ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

DOKTORI (PHD) ÉRTEKEZÉS

FRIED ZOLTÁN LÁSZLÓ

Termikus peremfeltételek becslése bio-inspirált optimalizálási eljárások alkalmazásával

Témavezetők:
Prof. Dr. Felde Imre
Prof. Dr. Szénási Sándor

**ALKALMAZOTT INFORMATIKAI ÉS
ALKALMAZOTT MATEMATIKAI
DOKTORI ISKOLA**

Budapest, 2024

Tartalomjegyzék

Nyilatkozat

Alulírott Fried Zoltán László kijelentem, hogy ezt a doktori értekezést magam készítettem és abban csak a megadott forrásokat használtam fel. Minden olyan részt, amelyet szó szerint, vagy azonos tartalomban, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Budapest, 2024.

Fried Zoltán

Köszönetnyilvánítás

Mindenekelőtt köszönettel tartozom családomnak, akik a munkám során támogattak és ösztönöztek, Mátyás fiamnak és Róza lányomnak, akik türelmesen viselték gyakori hiányomat.

Rengeteg köszönettel tartozom Prof. Dr. Felde Imrének, Prof. Dr. Szénási Sándornak és Prof. Dr. Tar József tanár uraknak, akik kitartóan egyengették a képzésem befejezéséhez vezető utamat. Köszönettel tartozom a Bánki Donát Műszaki Főiskola tanárainak az anyagtechnológia és gépészeti tanszékről, akik meggyőztek, hogy érdemes nekivágnom eme nemes feladatnak.

Köszönet illeti Réti Tamást, aki elindított a kutatói úton az egyetem alatt, és mentorom volt az azóta eltelt időben.

Nem utolsó sorban köszönettel tartozom Kovács Endre apósomnak, és feleségemnek, akik sasszemmel korrektúrázták ezt a dolgozatot.

1. Bevezetés, célkitűzés

A dolgozat az acélok bemerítéses edzése közben kialakuló hőátadási jelenség kvantitatív jellemzésére hivatott Hőátadási Együttható függvény számításához javasolt numerikus eljárások vizsgálatára fókuszál.

Az acélok bemerítéses edzése a hőkezelési eljárások közül az egyik leggyakrabban alkalmazott eljárás. A hipoeuktektoidos szerkezeti acélok általánosan alkalmazott immerziós edzési hőkezelése a munkadarab ausztenitesítéséből és az azt követő gyors hűtéséből áll. Ezen edzési művelet célja az elvárt tulajdonságú (szilárdságú és szívósságú) szövet létrehozása a munkadarab térfogatának előre tervezett hányadában. Az edzési művelet legkritikusabb része a munkadarab lehűtése az ausztenitesítési hőmérsékletről, amely döntően meghatározza a munkadarab szövetszerkezeti és mechanikai tulajdonságait. Az edzési folyamathoz használt hűtőközeg hűtési képessége a munkadarab és a hűtőközeg fizikai és kémiai tulajdonságaitól, valamint az egymáshoz viszonyított áramlási tulajdonságaitól függ. A hűtőközeg hűtési képességét a hőelvonásának a karakterisztikája írja le. A hűlési sebesség a teljes lehűlés alatt nem egyenletes, és bizonyos szakaszaiban viszonylag gyors ($> 100\frac{^\circ C}{s}$) is lehet. A hőkezelés tervezhetőségének, a termék végső tulajdonságainak biztosítása a hőelvonási karakterisztika ismeretének a záloga.

Az instacioner hőátadási jelenségek kvantitatív jellemzésére a hőfluxus mellett a „Hőátadási Együttható” (Heat Transfer Coefficient (HTC)) paraméter vagy függvény használatos. A HTC a hűtött test felületén időben és térben változó, a munkadarab által a környezetének átadott hőáram leírására alkalmas oly módon, hogy az magába sűríti számos, részleteiben nehezen modellezhető fizikai folyamat együttes hőtani hatását. A HTC függvény előállítása például az un. inverz hőátadási probléma (Inverse Heat Conduction Process (IHCP)) megoldásával lehetséges.

Kutatási célkitűzéseim az IHCP feladat megoldására alkalmas algoritmusok kidolgozására és vizsgálatára irányultak, amelyek az alábbi feltételeket elégítik ki:

- A fejlesztett algoritmus független a hőtani modelltől, végtelen hosszú tengelyszimmetrikus munkadarab kémiai tulajdonságaitól és átmérőjétől, valamint a hűtőközegtől.
- Eredménye az időtől vagy felületi hőmérséklettől függő HTC függvény, mint a munkadarab és a hűtőközeg közötti hőátadást jellemző függvény.

A dolgozatban 5 főfejezetben foglalkozom a probléma megoldásának lehetőségeivel, úgymint a grafikus gyorsítókártya, neurális hálózat, bio-inspirált algoritmusok, és gradiens alapú módszer alkalmazásának vizsgálatával, valamint ezen eljárások felhasználásával újfajta algoritmusokat mutatok be a célkitűzések megvalósítása érdekében.

Az a motiváció vezérelt a dolgozatban bemutatott új eljárások fejlesztése közben, hogy lehetőség szerint hatékonyság vagy pontosság tekintetében (a hozzáférhető információk alapján) felülmúlják a dolgozatban vizsgált hasonló elven működő, már létező megoldásokat.

2. A dolgozatban felhasznált elméleti módszerek áttekintése

Heat can never pass from a colder to a warmer body without some other change,
connected therewith, occurring at the same time.

Clausius statement of the second law of thermodynamics, 1854 [1]

2.1. Az optimalizálási probléma

A revese-engineering problémák megoldása számítógép igénybevétele esetén nagy valószínűséggel egy optimalizációs probléma megoldását jelenti, vagy végső soron arra visszavezethető. Legegyszerűbben egy optimalizálási problémát matematikailag a következőképpen lehet összefoglalni:

$$max\ f(\vec{x}) \text{ vagy } min\ f(\vec{x})$$
$$\vec{x} \in \mathbf{S} \subset \mathbb{R}^D$$

ahol az $f(\vec{x})$ a célfüggvény, az $\vec{x}$ egy D-dimenziós vektor a keresési térben, amely a megoldást reprezentálja, $\mathbf{S} \subset \mathbb{R}^D$ a célfüggvény értelmezési tartománya.

2.2. Hőátadás jelensége

A gyártási eljárások során az egyik leggyakrabban tapasztalt folyamat a hőátadás jelensége. A hőátadásnak három formáját különböztetjük meg attól függően, hogy az anyagok molekuláinak mozgását milyen fizikai jelenség írja le: a hővezetés (kondukció), a hőáramlás (konvekció) és a hősugárzás (radiáció).

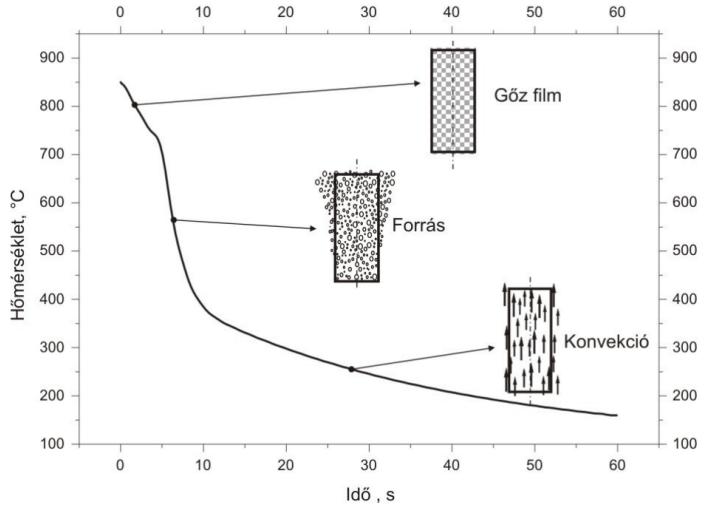
- Hővezetés esetén a közvetlenül érintkező anyagok elemi részecskéinek a mozgása közvetíti a hőt. A szilárd testekben a hő – néhány kivételtől eltekintve – hővezetés útján terjed.
- Hőáramlás esetén a hőt a folyadék vagy gáz elemi részeinek az áramlása, helyváltoztató mozgása továbbítja.
- Radiáció esetében a hőt a sugárzó test atomjainak mozgása során kibocsátott elektromágneses hullám továbbítja.

Hővezetéssel történő hőátadásnak elengedhetetlen feltétele az érintkező hővezető anyagok különböző pontjaiban fennálló hőmérséklet-különbség. A hővezetés következtében keletkező hőáram nagysága (a hőmennyiség rövid dt időegység alatt eső megváltozása) az anyagban a hőmérséklet-eloszlástól függ. Ebből következik, hogy hővezetés minden olyan esetben keletkezik, amikor az anyag egyes pontjai

között hőmérséklet-különbség lép fel, vagy ha két különböző hőmérsékletű anyag egymással érintkezik, vagyis egy fizikai rendszer energiát ad át egy másiknak. Az érintkezés hatására a nagyobb kinetikai energiával rendelkező molekulák energiájuk egy részét átadják a szomszédos, kisebb energiával rendelkező molekulának, amelynek hatására a hővezetés folyamata létrejön. Az egyensúlyi állapot akkor áll be, amikor a két érintkező anyag hőmérséklete annak pontjaiban időben állandósul. A dolgozatban csak a hővezetés esetét vizsgálom.

2.3. A hőátadás szakaszai

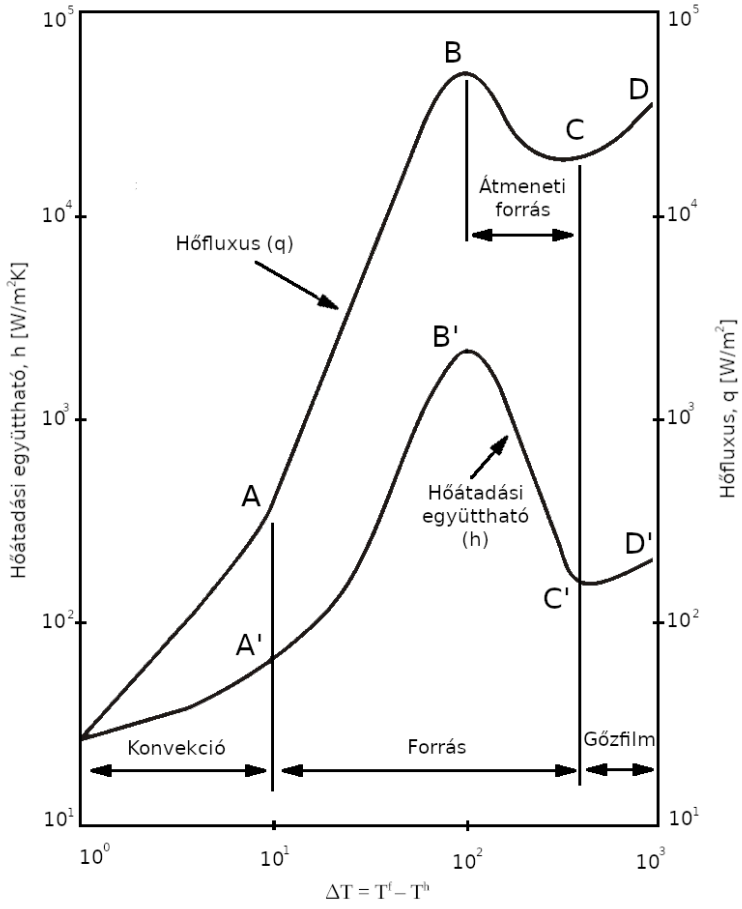
A hőkezelési eljárások során erősen szabályozott tranziens hőátadási folyamatok játszódnak le annak érdekében, hogy a kezelt alkatrész jellemzőit az előre meghatározott értékre állítsák be.



1. ábra. A folyadék hőelvonási szakaszai hengeres test palástja mentén [2]

Abban az esetben, ha a munkadarab felületi hőmérséklete sokkal magasabb, mint a hűtőközeg hőmérséklete [3], akkor – a megfigyelések szerint – a munkadarab hűtése során lejátszódó hőátadási folyamat négy különböző szakaszra bontható. Ez a négy szakasz rendre a gőzképződési vagy *gőzfilm*, az *átmeneti forrás*, a *forrás* és a *konvekciós* szakasz. Ezen négy szakasz egymástól általában jól megkülönböztethető, esetenként átfedés előfordulhat közöttük. Mivel az „átmeneti forrás” szakasza nehezen azonosítható, ezért gyakorlatban csak a *gőzfilm*, *forrási* és *konvekciós* szakaszokkal szokás foglalkozni (1. ábra). Minden egyes szakasz a lehűlés során létrejövő HTC és hőáramsűrűség különböző tartományait határozza meg. Ezek a szakaszok a Nukiyama diagramon (2. ábra) is megfigyelhetőek. A munkadarab hűtőközegbe helyezése után röviddel a felszínen gőzfázisú film képződik ($D - C$ szakasz), amely hőszigetelő rétegként viselkedik és lassítja a hőátadást. A hő ebben a fázisban sugárzás, vagy a filmrétegen keresztül történő konvekció útján terjed. A felület további hűlése során a gőzfilm réteg

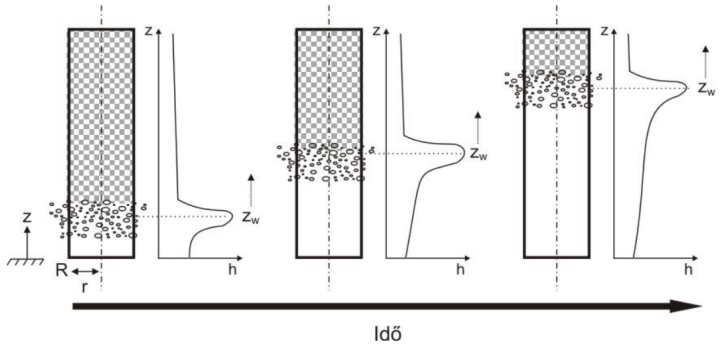
elvékonyodása tapasztalható, majd az úgynevezett „Leidenfrost” hőmérsékleten (*C* pont, amely a gőzfilm és az átmeneti forrás szakaszt választja el egymástól) a gőzfilm felszakadása után a hűtőközeg forrása kezdődik. Az átmeneti forrási szakaszban folyamatosan növekvő hőfluxus értéke ezen a hőmérsékleten éri el a maximumát (*B*) és ekkor kezdődik a tisztán forrási szakasz. A hőmérséklet további csökkenésével a hőáramsűrűség is csökken, egészen a hűtőközeg forráspontjáig (*A*), amellyel együtt a végső konvekciós szakasz is kezdetét veszi. Ebben a szakaszban – amely a forráspont alatti hőmérsékletet jelenti – játszódik le a felületi hőátadás [4–6].



2. ábra. A forrás szakaszai a közeg és a munkadarab felületi hőmérséklet különbségének függvényében (Nukiyama diagram [2])

A hűtőközeg forrási jelensége nagyon jól megfigyelhető egy hengerpalástú munkadarabon. A munkadarabnak a hűtőközegbe merítési pillanatától a merített felületi részén keletkezik a gőzfilm. Először a munkadarab henger palástjának aljáról indulva és felfelé haladva a munkadarab utoljára bemerített részéig (3. ábra). A gőzfilm hatására a hűtőközeg hőelvonó képessége a gőz képződésének helyén lecsökken, ugyanis a gőzfilm nagy „hőellenállásként” viselkedik. A hűtőközeg a munkadarabbal a gőzfilm hatására azzal nem érintkezik (nem nedvesít), ezzel lassítva a hőátadást. Azt a folyamatot,

amely leírja a gőzfilm (nedvesítési front) mozgását a munkadarab palástja mentén – a bemerítéssel ellentétes irányba a keletkezése és felszakadása között – nedvesítési kinetikának hívjuk. Szélsőséges esetekben előfordulhat, hogy a nedvesítési front mozgása a próbatest deformálódásához [7], esetleg károsodásához vezet.



3. ábra. A nedvesítési front (z_w) helyzete és mozgása [2]

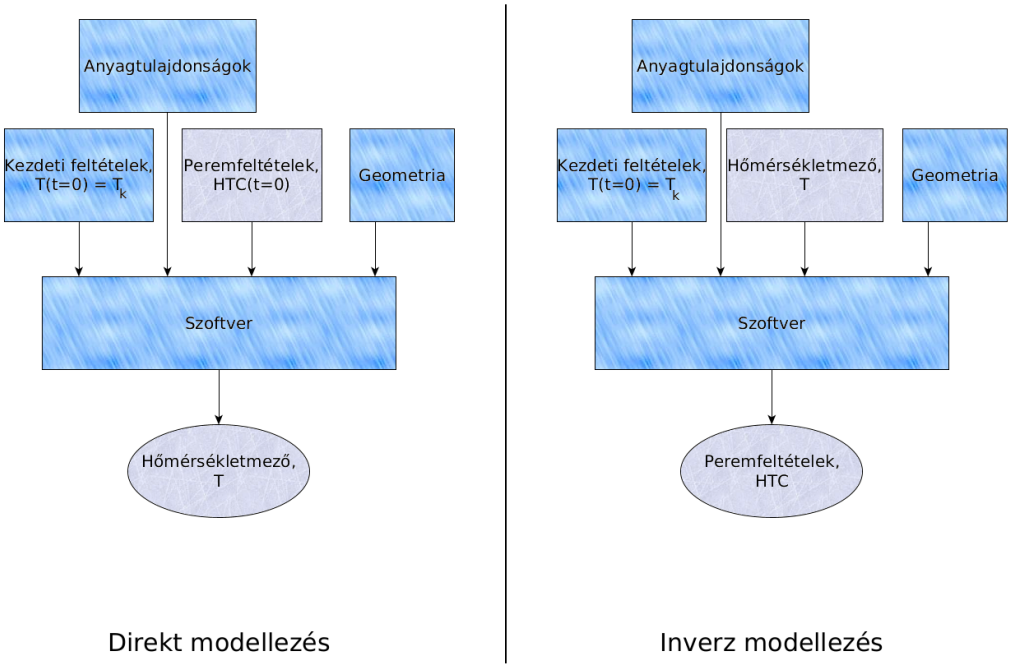
A nedvesítési front mozgását a HTC függvénnyel (h $[\frac{W}{m^2 \cdot K}]$) tudjuk leírni. A HTC függvény jellegét a munkadarab felületi hőmérséklete, felületének minősége, valamint a hűtőközeg hőmérséklete, áramlási viszonyai és a bázisát alkotó folyadék határozza meg.

2.4. A hőátadási együttható függvény és annak becslése

Egy hevítésnek, vagy hűtésnek kitett munkadarabban végbemenő hőmérséklet-eloszlás idő és hely szerinti változása a hővezetés Fourier-egyenletének megoldásával (1) határozható meg (hőátadás esetében), harmadfajú peremfeltétel mellett (2.5. fejezet). Mivel a munkadarabban és a munkadarab határfelületén végbemenő hőközlési folyamat nemlineáris (függ a munkadarab fizikai tulajdonságaitól (hővezetési tényező, fajhő, stb.) és a HTC-től, a HTC pedig az időtől és helytől), ezért a Fourier-egyenlet megoldása zárt alakban nem adható meg. Emiatt numerikus eljárást kell alkalmazni.

Abban az esetben, amikor a termikus kezdeti és peremfeltételek egyértelműen adottak vagy meghatározhatóak, „helyesen feltett” (ún. „well posed”) problémáról beszélünk, a numerikus módszerek alkalmazása nem ütközik nehézségbe. Az ilyen típusú modellezési eljárást „direkt” modellezésnek hívjuk. A direkt modellezés esetén az okból (hőfluxus) határozzuk meg az okozatot (hőmérséklet). Más szavakkal, a „direkt” modellezés során– azaz a hűtött munkadarabban kialakuló hőmérséklet-eloszlás időtől való függésének – az adott kezdeti eloszlás és a határfeltételek, valamint a HTC felületi értékeinek, azaz a $h(\vec{r}, t)$ (ahol t $[s]$ az időt, $\vec{r}$ pedig a munkadarab felületének egy pontját jelöli) függvény birtokában, a feladat egyértelmű megoldással rendelkezik.

A termikus peremfeltétel becslésére a hőmérsékletmező ismeretében van lehetőség, és ekkor az okozatból (kialakult hőmérsékletmező) kell az okra (termikus peremfeltétel) következtetni. Ebben az esetben az inverz modellezési feladattal állunk szemben, amely a mért (hőmérsékletmező) hőmérsékletgörbék ismeretét, valamint a modellezni kívánt munkadarab hőtani-szimulációs modelljének meglétét feltételezi. Az ok, mint a hőáram predikciója, egy hiányosan definiált, „rosszul feltett” (ún. „ill-posed”) feladat a hőátadási probléma erős nem-linearitása miatt. (Ezek a matematikai problémák az un. „reverse engineering” problémák közé sorolandók, amelynek adott feltételek mellett nem létezik egyedi, unikális megoldása. [8].)

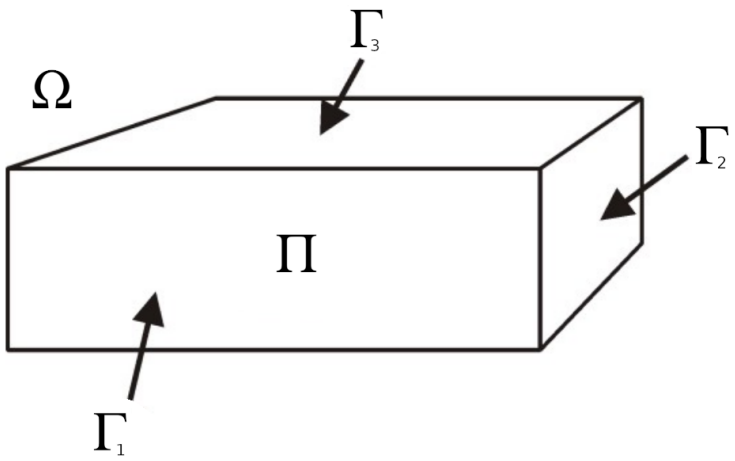


4. ábra. A direkt és az inverz modellezés elvi ábrája

Matematikai szempontból a probléma egy optimalizálási feladatként fogalmazható meg, amelyben egy feltételezett kiindulási $h(\vec{r},t)$ eloszlás mellett a munkadarab egyes pontjaiban adott hőmérséklet-értékek időfüggésének mért és számított eltérését kell minimalizálni a $h(\vec{r},t)$ eloszlás változtatásával [8–14].

2.5. A hőátadás matematikai modellje

A hőcsere-folyamatok során egy, a munkadarabban kialakuló hőmérsékletmező számításához a hőátadás Fourier egyenletének megoldására van szükség. A dolgozat fókuszában lévő inverz matematikai módszer vizsgálatához egy tengelyszimmetrikus (hengeres) geometriára dolgoztam ki a hőtani modellt.



5. ábra. A hőátadási folyamatban résztvevő test

Egy Γ_i, $i \in \{1, 2, 3, \ldots\}$ felületekkel határolt Π test és az azt körülölelő Ω közeg (5. ábra) közti hőátadási folyamatot a hővezetés Fourier differenciálegyenlete írja le:

$$\nabla(\lambda(\vec{r}, T) \cdot \nabla T) + Q(t, \vec{r}, T) = \rho(\vec{r}, t) \cdot C_p(\vec{r}, T) \cdot \frac{\partial T}{\partial t} \tag{1}$$

ahol t $[s]$ az idő, $\vec{r} \in \Pi$ a helyvektor, $T = \tilde{T}(\vec{r}, t)$ $[°C]$ a Π test hőmérséklete, $Q(t, \vec{r}, T)$ $[J]$ a szilárd fázisú átalakulás következtében képződő hőmennyiség, $\lambda(\vec{r}, T)$ $\left[\frac{J}{s \cdot m \cdot K}\right]$ hővezetési tényező, $\rho(\vec{r}, t)$ $\left[\frac{kg}{m^3}\right]$ a sűrűség, $C_p(\vec{r}, T)$ $\left[\frac{J}{kg \cdot K}\right]$ a fajhő. A $t = 0$ kezdeti időpillanatban a hőmérsékletet a

$$T(\vec{r}, t = 0) = T_0(\vec{r}) \tag{2}$$

egyenlet adja meg, ahol T_0 a Π test kezdeti hőmérséklete, valamint a felületeihez tartozó peremfeltétele az

$$-\lambda \frac{\partial T}{\partial t} = \lambda_i(T'(\vec{r}', t) - T_q) \tag{3}$$

egyenlet, ahol $\lambda_i(T')$ a Γ_i a felületekhez tartozó HTC $i \in \{1, 2, 3, \ldots\}$, T' a felület hőmérséklete, $\vec{r'} \in \Gamma_i$ és a T_q az Ω közeg hőmérséklete.

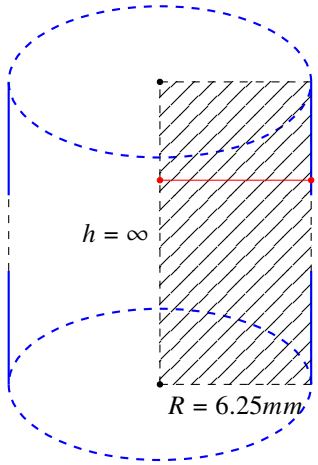
A hőátadási folyamatot végtelen hosszú, de véges átmérőjű henger esetében, azaz csak sugárirányú, $1D$ hővezetést feltételezve, a (4) egyenlettel jellemezhetjük:

$$\frac{\partial}{\partial r}\left(\lambda \cdot \frac{\partial T}{\partial r}\right) + \frac{\lambda}{r}\frac{\partial T}{\partial r} + q_v = \rho C_p \frac{\partial T}{\partial t} \tag{4}$$

ahol t az idő, r a hely koordináta, a $T(r, t)$ a t-től és r-től függő hőmérséklet, q_v a látens hőmennyiség, ρ a sűrűség, C_p a fajhő és λ a hővezetési tényező. A dolgozatban csak az $1D$ hővezetést vizsgálom.

2.6. A geometriai, fizikai modell

A dolgozatban használt geometriai modellt a 6. ábrán mutatja.



6. ábra. A geometriai modell

A feladat komplexitása nagymértékben csökkenthető speciális alakú munkadarabok és speciálisan elhelyezett, a hőmérséklet mérésére a munkadarabon belül kijelölt pontok alkalmazásával, továbbá olyan anyag(ok) használatával, amely(ek)en belül sem olyan kémiai folyamatok, sem fázisátalakulások nem történnek, amelyek jelentős hőt termelnének vagy nyelnének el, sem pedig az anyag hővezetési tulajdonságait jelentős mértékben nem befolyásolnák. Ilyen tulajdonságokkal rendelkezik például az ISO 9950 szabványban javasolt ötvözet [15]. Az Inconel 600 ötvözetről a hűlési vizsgálatok hőmérséklet-tartományában jól és pontosan ismert termikus adatok állnak rendelkezésre [16]. A választott anyag tulajdonságai miatt a vizsgált tartományban látens hő termelése/elnyelése nem következik be. Az Inconel 600 termikus adatai [16]-ből származnak. Az anyag sűrűsége közel konstans: $\rho = 8420 \left[\frac{kg}{m^3}\right] \equiv const.$

2.7. A hűlési folyamat modellje és annak végeselemes közelítése 1D hengerszimmetrikus munkadarabra

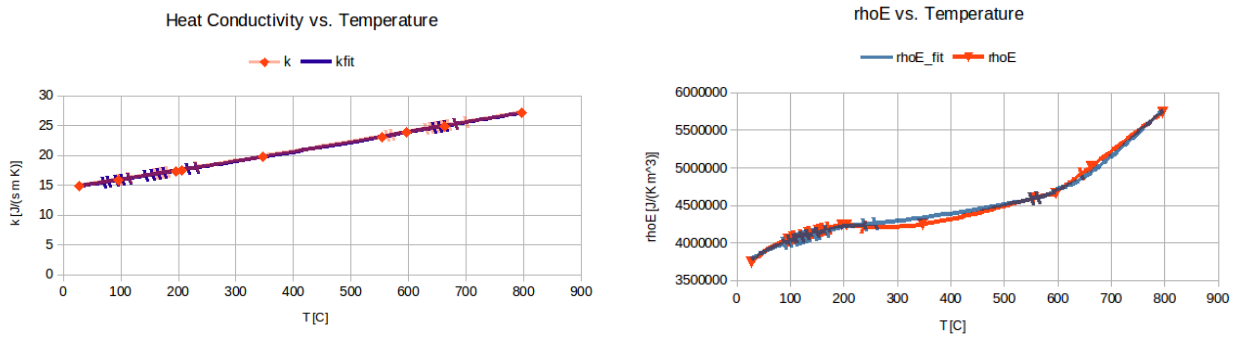
„Hosszú, henger alakú munkadarab” esetén szimmetria-megfontolások alapján a munkadarab tengelyének irányába nem, de hengerszimmetrikus energiaáramlást feltételezve egyváltozós hőáramlási egyenletté redukálhatóak a számítások. A választott anyag tulajdonságai miatt a hővezetési egyenlet leegyszerűsödik az (5) egyenlet szerint:

$$\frac{\partial}{\partial r}\left(\lambda(T(r,t))\frac{\partial T(r,t)}{\partial r}\right) + \frac{\lambda(T(r,t))}{r}\frac{\partial T(r,t)}{\partial r} = \rho C_p(T(r,t))\frac{\partial T(r,t)}{\partial t} \quad , \tag{5}$$

amelyben a $t\ [s]$ változó az időt jelöli, $r\ [m]$ pedig a középvonaltól mért sugarat, ami a probléma másik független változója, $T\ [^\circ C]$ pedig a hőmérséklet. A „hővezetési tényező” $\lambda(T)\ \left[\frac{W}{m \cdot K}\right]$ és az

állandó nyomáson vett fajhő $C_p(T)$ $\left[\frac{J}{kg \cdot K}\right]$. A hőmérséklet harmadrendű polinomiális függvényeként közelíthető a $T \in [27 - 796{,}45]$ $[°C]$ tartományban a 7. ábra szerint.

A vizsgálandó munkadarab geometriai és termikus tulajdonságainak jellemzésére felírt egyenletek egyszerűsítése mellett a hűtőfolyadék viselkedése további jelentős komplexitásforrást jelent az egyenletekben. Az így keletkező nagy számítási kapacitások az evolúciós technikákat, és Graphical Processor Unit (GPU) felhasználását is indokolttá teszi. [17–21].



(a) (b)

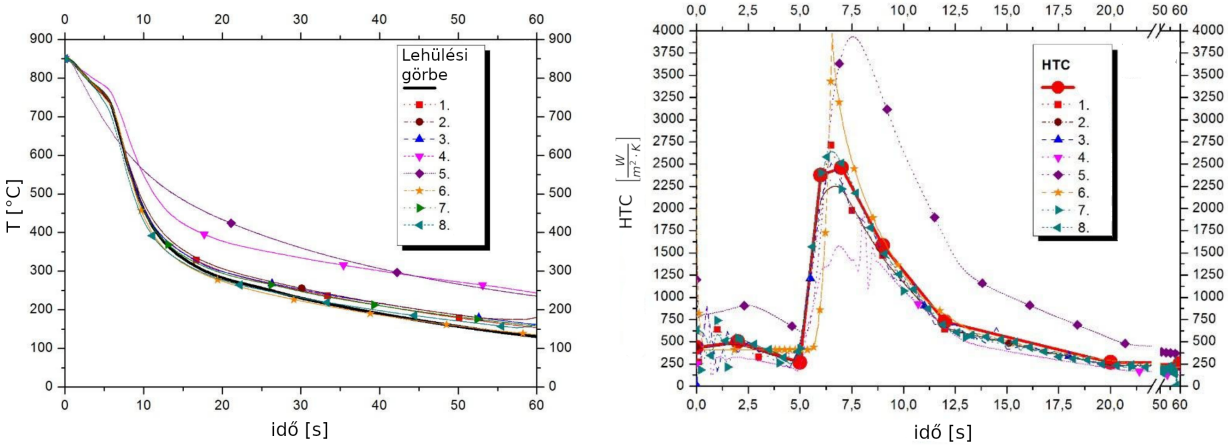
7. ábra. A „hővezetési tényező” (a) ábra) és a $\rho_E \equiv \rho C_p$ mennyiség (b) ábra) hőmérséklettől való függése az (5) egyenletben: harmadrendű polinomok az Inconel 600 táblázatos adataira illesztve [16] alapján

Az 5. egyenletet megfelelő *határfeltételek* egészítik ki R sugarú munkadarab esetében a (6) szerint:

$$-\lambda(T(R,t)) \left.\frac{\partial T(r,t)}{\partial r}\right|_{r=R} = h(t)(T(R,t) - T_q) \ , \tag{6}$$

amelyben T_q $[°C]$ a hűtőfolyadék „bulk” anyagának hőmérsékletét jelöli turbulens áramlás esetén, és $h(t)$ $\left[\frac{W}{m^2 \cdot K}\right]$ jelöli a folyadék határrétegében a HTC függvényt, amelyről feltételezem, hogy a munkadarab hűlésével időben is változik. Meg kell még adni továbbá a kezdeti feltételeket, amelyeknek kompatibilisnek kell lennie a határfeltételekkel. E kompatibilitás a végeselem közelítésre ró ki világos feltételeket, amelyeket a 7.1. alfejezetben részletezek.

Annak érdekében, hogy „realisztikus” becslést adhassak a problémára, az irodalomból vettem át kvalitatív és közelítő kvantitatív adatokat a 8. ábrán látható mintára. A kezdeti egyenletes hőmérséklet $T_{ini} = 850$ $[°C]$ kissé magasabb volt, mint a [16] referenciában adott maximális adat. A kis eltérés miatt ebben az esetben az illesztett polinomokat nemcsak „interpolációra”, hanem „extrapolációra” is felhasználtam.



(a) A [22] irodalom 3. ábrája

(b) A [22] irodalom 4. ábrája

8. ábra. Hűlési görbék és HTC adatok különböző közelítési módszerekkel számolva a választott munkadarab esetén

2.8. A célfüggvény

2.8.1. A célfüggvény általános megfogalmazása

A célfüggvény megfogalmazható úgy is, mintha egy skalár értékű függvényt kellene minimalizálni k darab független változójának változtatásával. E skalár érték lehet például a $T^c \in \mathbb{R}^L$, $L \in \mathbb{N}$ is, ami a munkadarab adott tengelyében mért hőmérséklet különböző időpontokhoz tartozó értékeiből áll össze, mint egy véges tartományon bevezetett diszkrét rácson és egy Frobenius normával definiálva: $S \overset{def}{=} \|T^c - T^m\|$.

A további vizsgálatokra egy tetszőleges $\nu > 0$ és $\nu \in \mathbb{N}$ paraméter bevezetésével az előbbi célfüggvény

$$S \overset{def}{=} \|T^c - T^m\|^\nu \tag{7}$$

általánosítása szintén megfelelő célfüggvényre vezet, amely szintén növekszik a hibával, de másképp, mint az eredeti célfüggvény, s ugyanúgy annak 0 értéke felel meg a globális optimumnak.

A (7) egyenlet két speciális esete, amikor $\nu = 1$ és $\nu = 2$. Jelen kutatásban a $\nu > 2$ esetek nem jelentenek előnyt. Ha $\nu = 2$, akkor egy elterjedt célfüggvény alakját kapjuk [23], ugyanis az $S = 0$ pontban differenciálható, és ha $S > 1$, akkor az S célfüggvény értékét a végtelen felé tolja, vagyis széthúzza a tartományt, viszont ha $0 < S < 1$, akkor a nulla felé sűríti a kapott célfüggvény értékeket. Ez azt jelenti az algoritmus szempontjából, hogy a nagyobb eltéréseket a referenciától hatványozottabban bünteti, az 1-nél kisebb eltéréseket pedig erősebben jutalmazza. Amennyiben $\nu = 1$, akkor az abszolút hibafüggvényhez jutunk, amikor minden eltérés a referenciától egyenlő mértékben számít bele a hibafüggvénybe. Ez az eset a 0 pontban nem differenciálható.

Korunk számítógépei a lebegőpontos számábrázolás esetén az IEEE 754 szabvány szerint működnek [24], ami többnyire azt jelenti, hogy nagy és (esetünkben) kicsi számok számábrázolása esetén egyre nagyobb eséllyel nem lehetséges pontosan a kívánt számot ábrázolni. Ezt a korlátot a GPU alkalmazása esetén kell komolyabban figyelembe venni.

2.8.2. Dolgozatban vizsgált heurisztikus eljárások célfüggvénye

A 2.8.1. fejezetben ismertetett információk tudatában az (1) differenciálegyenletnek a (3) peremfeltételét kielégítő megoldásának megtalálásához mégis a (8) célfüggvényt választottam.

$$S = \sum_{k=1}^{K} (T_k^m - T_k^c)^2 \rightarrow \min \tag{8}$$

A dolgozatban a $S = 0$ pont azt a pontot jelenti, amikor megvan a keresett függvény, és befejeződnek a számítások. Mivel a célfüggvény abszolút 0 értékének elérése egyelőre nem cél, a fejlesztett szoftver önkényesen megemeli az elérendő célt egy ε^* értékkel. A ε^* tudatos megválasztásával a lebegőpontos számok ábrázolásának problémája az esetek többségében megfelelően kezelhető.

ahol a T_k^m a mért, a T_k^c a számított lehűlési görbe hőmérsékletértékét jelenti a $k \in \{1, 2, 3, \ldots K\}$ minden iterációban, S a célfüggvény, a K pedig a számolt lehűlési görbe pontjainak a száma. Ezzel az inverz hővezetési probléma megoldását visszavezettem egy minimumkeresési problémára. Amennyiben

$$\lim_{I \rightarrow \infty} S = 0 + \varepsilon \tag{9}$$

ahol az $I \in \mathbb{N}$ az iteráció számát mutatja, amelyet ideális esetben nem kell előre definiálni, mivel léteznek olyan leállási feltételek, amelyekkel elérhető, hogy az algoritmus ne fusson végtelen ideig. Ezért a numerikus algoritmusnak a futása akkor ér véget, ha a számított és a mért lehűlési görbe között a legkisebb a különbség, vagyis a (9) célfüggvény minimum értékét kell meghatározni, és ekkor meghatározásra kerül egy lehetséges megoldás is. Az $\varepsilon \geqq \varepsilon^*$ egy olyan választott kis érték, amely elérésekor az eredménygörbe már egy előre meghatározott hibával rendelkezhet.

2.8.3. Célfüggvényhez köthető kényszerfeltételek

A műszaki alkalmazások területén, mint például az optimális szabályozások esetében is, gyakorta fordul elő valamilyen célfüggvény kényszerfeltételek melletti minimalizálása.

Az optimális szabályozások alapfeladata, hogy gyakorta egymásnak ellentmondó feltételek közt kompromisszumos megoldást kell találni úgy, hogy a feladatban vannak „keményen”, azaz precízen

betartandó követelmények is. A „puha”, vagyis kompromisszumosan betartható feltételek megfogalmazhatóak, mint nemnegatív „költségeket” kifejező járulékok összegéből összeállított minimalizálandó „költségfüggvény”, míg a „kemény” feltétel azt fejezi ki, hogy a szabályozott rendszernek van egy dinamikai modellje, amelytől eltérően az nem tud működni, ezt tehát mindig precízen kell figyelembe venni a költségfüggvény minimalizálása közben. Ez utóbbi feltételeket, mint „kényszerfeltételeket” szükséges matematikailag megfogalmazni.

Ha a feladatokat matematikailag a variációszámítás eszköztárával, funkcionálok minimalizálásával kell kezelni, a Klasszikus Mechanika variációs elvekkel való megfogalmazásával nagyon szoros analógiát mutató *„Hamilton – Jacobi – Bellman egyenletek”* kerülnek elő, s ezek numerikus megoldására a rendkívül erőforrás-igényes *„dinamikus programozás”* módszeréhez jutunk [25, 26].

E megoldás matematikailag tovább egyszerűsített változata, amikor valamilyen véges horizonton diszkrét időrácson számolom ki a célfüggvényeket, és a dinamikai modellt is ezen közelítem. Ekkor a véges sok pontra közvetlenül alkalmazható a Lagrange Redukált Gradiens módszere. Minél inkább szűkítjük a költségfüggvényeket, a matematikai kényszeregyenletek formájának, és a használt dinamikai modellek matematikai formájának megengedett körét, annál több, többé–kevésbé általános megállapítás tehető a várható megoldásról.

Abban az esetben, ha csak a kényszerfeltételek lineáris megfogalmazásához ragaszkodom, és figyelembe veszem, hogy pontosan annyi Lagrange-szorzó létezik, mint ahány $\dot{x}$ komponens, a Klasszikus Mechanikai Hamilton-féle kanonikus mozgásegyenletekkel [27, 28] analóg egyenleteket kapunk, amelyben a „mozgás” folyamán megmarad egy mesterséges „energia” függvény (a Hamilton-függvény), továbbá az x komponensek és a Lagrange szorzók mint *kanonikus változó-párok* jelennek meg, amely gyakran instabil megoldáshoz vezet.

Ezért két lehetőség adódik: vagy figyelni kell a szűkített költségfüggvények számára és funkciójára (például a dolgozatban a (8) célfüggvény és az 5.4. fejezetben ismertetett másodlagos célfüggvény alkalmazása), vagy ha kihasználjuk a kvadratikus költségfüggvényből adódó egyszerűsítési lehetőségeket a kényszerfeltételek átalakíthatóak lineáris egyenletekké, és így már csak egy viszonylag leszűkített probléma-osztály numerikus kezelésével kell foglalkozni [29] (például a 7. fejezetben bevezetett kísérlet).

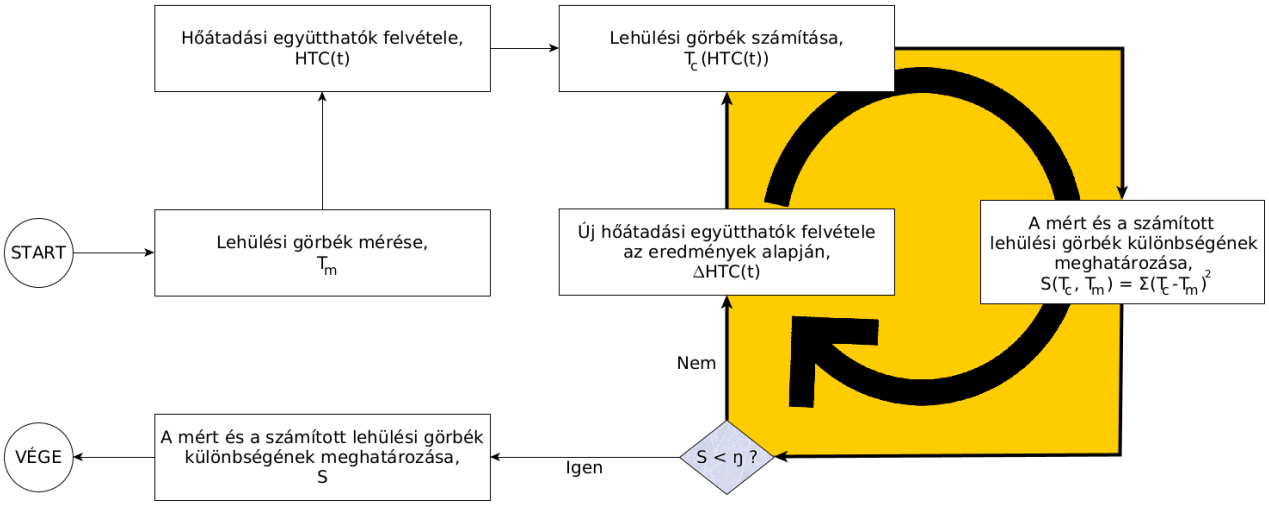
2.9. Az inverz hővezetés számítás elve

Az inverz hőtani modellezés során a termikus peremfeltételek becsléséhez egy munkadarab adott pontjában a mért és a számított hőmérséklet függvények ismeretében a HTC függvény paramétereinek meghatározása optimalizálási feladat. Azokat a HTC függvényértékeket keressük, amelyek alkalmazása

mellett a számított lehűlési görbék a legjobban közelítik a mért lehűlési görbéket. Ennek megfelelően a célfüggvény (8) minimumánál adódik a görbék legkisebb eltérése.

A termikus határfeltételek becslésére javasolt eljárás lépéseit a 9. ábra szemlélteti. A HTC függvény becslése a következő lépésekből áll:

1. A lehűlés során a vizsgált munkadarab adott pontjaiban a lehűlési görbéket ($T_i^m, i \in \{1, 2, 3, \ldots, K\}$) felvesszük a $[t_{\text{ini}}, t_q]$ intervallumba (a K a számolt lehűlési görbe pontjainak a száma).
2. A HTC függvények kezdeti értékeit a keresési tartományban véletlenszerűen megválasztjuk a $[t_{\text{ini}}, t_q]$ intervallumban ($h_i(t), i \in \{1, 2, 3, \ldots, K\}$).
3. A lehűlési görbék számítását ($T_i^c, i \in \{1, 2, 3, \ldots, K\}$) $h_i(t)$ alapján elvégezzük.
4. A mért és számított lehűlési görbéket összehasonlítjuk egymással és a célfüggvényt kiszámítjuk.
5. Ha a különbség a megadott tolerancia határon kívül esik, akkor módosítjuk a $h_i(t)$ értékeit a választott optimalizációs algoritmus alapján és folytatjuk a 2. lépéssel.
6. Ha a közelítés megfelelő, akkor a számítás végeredménye – a teljes hőmérséklet ciklus ismeretében – a megválasztott HTC függvény lesz, mint a felületi hőmérséklet függvénye $h_i(T)$.



9. ábra. Az inverz algoritmus lépései

Az inverz analízis célja, hogy iteratív becslést adjon az ismeretlen $h(t)$ függvényre a választott eljárás alkalmazásával. A célfüggvény értékét az n. iterációban ($n \in \{0, 1, 2, \ldots\}$) a számított és mért hőmérséklet-görbék különbségét a (8) egyenlet felhasználásával határoztam meg. Az algoritmusnak garantálnia kell, hogy az optimumtól eltérő esetekben a számított és a munkadarab adott helyén mért hőmérséklete között könnyen elhanyagolható hőmérséklet-különbség adódjon.

2.10. Inverz problémák megoldhatóságának korábbi eredményei

Az inverz problémák matematikai megoldásához kimunkált kezdeti eljárások és formulák elsősorban az időjárás-előrejelzésre és földkéreg hővezető-képesség becslésre irányultak (Fourier, Poisson és Kelvin munkái [30]). A tranziens hőátadási folyamatok közben kialakuló hőfluxus meghatározására már a múlt század közepétől dolgoztak ki algoritmusokat (többek között Stolz, Beck, Aldoshin, Golosov, Zhuk, Alifanov és Stefan [31–33]), a hőátadási problémák megoldására pedig a Cauchy-féle általánosított formulát dolgozták át végtelen hatványsor formájára. Ezt az eljárást tekinthetjük a konstans együtthatók megoldási eljárása mellett az egydimenziós IHCP első egzakt megoldásának, annak ellenére, hogy ez a módszer Burgraf [34] és Tyomkin [35] munkáinak köszönheti a népszerűségét. Az IHCP megoldására irányuló kutatások az 1950-es éveket követően kaptak lendületet, ugyanis az orosz és az amerikai űrprogramokhoz, valamint az atomerőművek reaktortartályaiban végbemenő hőcsere számításához jelentős számú kutatóra volt szükség. Stolz [36] például az egyszerű geometriájú testek vízben hűtésekor a felületi hőfluxus számítására kidolgozott eljárását 1960-ban publikálta. A témában igazi áttörést először Tikhonov 1963-ban [37], majd [38, 39] publikált eredményei hoztak, amelyek nemcsak a fizikai jelenségek matematikai leírásában, hanem a feladat megoldásához készített programok felépítésében is új távlatokat nyitottak. Az általa kidolgozott regularizációs eljárást, valamint annak továbbfejlesztett változatait számos tudományterületen alkalmazták sikerrel. Ezek miatt az inverz hőátadási problémák megoldására kidolgozott univerzális matematikai módszert iteratív regularizációs technikának is hívjuk. Ennek az eljárásnak egy új megközelítése az Alifanov [40] által publikált nem-lineáris gradiens módszer alkalmazására épül. A számítástechnika további fejlődésével az 1970-es évek elején a „heurisztikus regularizáció”-nak nevezett megközelítés kezdett népszerűvé válni, majd Kozdoba [41] a „trial and error” nevű módszerrel, míg Matsevity és Symbirsky [42] a „digitális dinamikus szűrő” eljárással kísérletezett. Ezek a megoldások különböző direkt matematikai módszerek szisztematikus használatán alapulnak, és különféle gyakorlati esetekre megfelelő számítási stabilitás mellett meglepően pontos eredményre vezetnek. Beck [43] és Tikhonov [44] munkáinak köszönhető az IHCP kezelésére elsőként kidolgozott matematikailag alátámasztott, pontos, könnyen algoritmizálható eljárás, amely a hőátadási probléma linearizációjára és regularizációs eljárások alkalmazására épül. A véges differencia (Finite Difference Method (FDM)) és véges-elem (Finite Element Method (FEM)) módszerek alkalmazásával Alifanov [45] számos nem-lineáris IHCP eset megoldásához nyitotta meg az utat. Az elmúlt évtizedekben több ezer publikáció született az inverz hőátadási problémák témakörében.

2.11. Bio-inspirált módszerek

A bio-inspirált módszerek a természeti folyamatokat veszik alapul az egyes természeti folyamatokat vezérlő szabályok keresésekor. Az alapötlet az, hogy egy optimalizációs feladathoz választott egyszerű lépéssorozat közé a véletlent, mint vezérlő tényezőt beemelve az egyenletekbe, meglepően jobb eredmény kapható, mint ha minden paraméter értékét szigorú irányítás alatt tartanánk. A bio-inspirált algoritmusok a genetikus algoritmusokat egy csoportja.

A genetikus algoritmusokat [46, 47] sikeresen alkalmazzák különböző típusú inverz hőátadási problémák megoldásához. A különböző numerikus optimalizációs technikák kvantitatív jellemzése megmutatta [17], hogy a sztochasztikus módszerek gyorsabbak az IHCP megoldásakor, mint a komplex termikus peremfeltételek felhasználásával működő gradiens módszerek.

A számítási intelligenciát használó eljárások az elmúlt 30 évben komoly fejlődésen mentek keresztül. A fejlődési irányok egyre szerteágazóbbak, mivel egyre több igény jelentkezik a komoly és hatékony algoritmusokra, mint például az élelmiszerek hővezetési tulajdonságainak meghatározása [48].

A természetben működő – általában hatékony – decentralizált biológiai mechanizmusokra a robusztusság, skálázhatóság, adaptivitás és önszervező készség jellemző. Ezen tulajdonságokat a raj-intelligencia szemléletével megalkotott algoritmusok igyekeznek magukba olvasztani és azokat hordozni, valamint lehetővé tenni komplex optimalizálási feladatok hatékony megoldását. Ilyen algoritmus például az Ant Colony Optimization [49], Particle Swarm Optimization [50], Artificial Bee Colony Optimization [51], vagy a Fireworks Algorithm [52].

A dolgozatban kétfajta bio-inspirált algoritmussal foglalkozom : [53, 54], azokat későbbi fejezetekben részletesebben is ismertetem.

2.12. A számított hőátadási együttható függvény közelítésének pontossága

A általam kifejlesztett inverz számítási módszer vizsgálatát numerikus szimulációs szoftvereken keresztül végeztem. A szakirodalmi források ajánlásainak megfelelően vizsgáltam [10, 11, 43] a számított HTC függvény közelítésének pontosságát a mért görbéhez viszonyítva. Az eredményeket az adott algoritmus tárgyalásakor a 4.-7. fejezetekben mutatom be.

2.13. Az alkalmazott hardver és szoftver elemek

A számításokat egy átlagos teljesítményű Dell laptop segítségével végeztem. Ennek az operációs rendszere egy Ubuntu Linux. A számítógépbe NVIDIA T1200 GPU kártya gyárilag került, melyet csak a neurális hálózat számításaihoz, a szükséges lehűlési görbék és célfüggvény értékek eljárásainak

számításához használtam. A szimulációs szoftver az Firworks Algorithm (FWA), Particle Swarm Optimization (PSO) algoritmusokhoz C++11 nyelven, a neurális hálózatot a TensorFlow keretrendszer segítségével építettem fel, amelyet Python nyelven implementáltam. A számítási eredmények grafikonjainak elkészítéséhez a Julia [55] és Python programnyelveket alkalmaztam. Jelen dolgozatot Latex nyelven írtam. A dolgozatban használt fontosabb szoftverek verzióját és paramétereit a 9. melléklet tartalmazza.

3. Grafikus gyorsítókártya alkalmazása a hőmérséklet-eloszlás számításának gyorsításához

A Graphical Processor Unit (GPU)-t eredetileg számítógépes játékok grafikai elemeinek megjelenítéséhez szükséges számítások gyorsításához fejlesztették ki – ugyanis az élethűbbnél élethűbb környezet képi ábrázolása rengeteg, speciális számítási kapacitást igényel. Ezt az igényt a párhuzamosan számolható grafikai elemek – előbb csak a 2D-s, majd később 3D-s is – renderelésének párhuzamosítása (gyorsítása) – segítségével érték el. Ennek eredményeként a képek egyre realisztikusabb és gyorsabb megjelenítését tette lehetővé. Az esetek többségében mindenfajta matematikai programozhatóság nélkül, a megjelenítés technológiájának gyorsításával. Ez a speciális számítási kapacitás keltette fel a mérnökök figyelmét, és matematikai számítások elvégzésére kezdték használni a grafikus kártyákat. Ebben az időben a szűkös parancskészlet és a matematikai műveletekhez szükséges utasítások hiánya miatt grafikai parancsokat használtak különféle műveltek elvégzésére, kihasználva ezen műveletekhez tartozó parancsok esetlegesen párhuzamos végrehajtási lehetőségét. Az első valódi programozhatóságot az Open Graphics Library (OpenGL) és a DirectX Application Programming Interface (API)-k megjelenése és támogatása hozta, ami eleinte csak a grafikus elemek és környezeti hatások szimulálásához szükséges minél egyszerűbb elemek szabad alkalmazhatóságából állt. Az első próbálkozások a GPU-k nem grafikus számításokhoz történő felhasználására a grafikus API-k corner case eseteinek használatát jelentették. Ráadásul csak korlátozott hardvertámogatás volt az általános célú programozáshoz, de ennek ellenére nagy számítási idő csökkenéseket lehetett elérni [56]. Ezek a kezdeti sikerek mégis kevésnek bizonyultak. A grafikus processzorok nem grafikus felhasználása miatt a mérnökök arra kérték a gyártókat, hogy adják hozzá a matematikai elemek hardver és szoftver támogatását is a GPU-khoz. Az egyre növekvő felhasználói igény felismerése után a kártyák gyártói felkészítették kártyáikat a szükséges matematikai műveletek kezelésére is. Ez lehetővé tette a szélesebb felhasználási igények kielégítését, és elérkezett az General Purpose Graphical Processor Unit (GPGPU)-k kora, mint például [57–60].

Az Nvidia Compute Unified Device Architecture (CUDA) [61, 62], az Advanced Micro Devices (AMD) Close To Metal (CTM), valamint a nyílt szabványú Open Computing Language (OpenCL) [63] megoldásai révén a GPU hardver képessé vált az általános célú számítások támogatására, valamint megszületett a masszívan több szálú hardver magas szintű programozási felülete. A programozók külön absztrakciót kaptak a GPGPU memória címterületéhez, ahol az adatokat tárolják a párhuzamos eljárások számára. A GPGPU programozható felülete elsősorban a C nyelvhez készített kiterjesztéseken

keresztül volt elérhető, de egyéb nyelvekhez is elkészültek támogatási modulok. Idővel általános, gyakran használt alapvető matematikai műveletekhez, eljárásokhoz számos programkönyvtár készült a hozzáértő mérnökök és programozók hozzájárulásaként annak érdekében, hogy a GPGPU programozásának területén kevésbé képzett kutatók számára is hozzáférhetővé váljon ez a technológia [64]. Ezen lehetőségek a GPGPU-k szélesebb körű elfogadásához és használatához vezetett. A továbbiakban a GPU rövidítés alatt a GPGPU-kat értem.

3.1. A szimulációkhoz használt grafikus gyorsítókártya

A GPU-t a lehűlési görbék, a görbékből számítható célfüggvény értékek számításához, valamint a neurális hálózat számításához használtam [65, 66] a 4, az 5 és a 6. fejezetekben. Az általam használt GPU kártya az Nvidia T1200 kártyája volt. Az Nvidia GPU kártyáinak a fejlettségi szintjét a kártya Compute Capability értéke határozza meg, ennek a kártyának a Compute Capability értéke 7,5, miközben a legújabb Nvidia grafikus kártyák Compute Capability értéke már 9. A használt kártya a kitűzött feladat megoldhatóságának demonstrálásához elegendő. A feladat megoldásához az Nvidia grafikus kártyát választottam, mert a programozáshoz használt API (CUDA) egy viszonylag elterjedt programozási környezetet biztosít. Mindemellett a választásom teljesen szubjektív.

3.2. A választott GPU kártya főbb paraméterei

Jelenleg a kereskedelmi forgalomban otthoni felhasználásra kapható Nvidia GPU kártya Computer Processor Unit (CPU)-inak bitszélessége többnyire 32 bit. Ez egy nagyon fontos információ annak a tekintetében, hogy ez a kártya minden működési paraméterét alapvetően meghatározza. A kártya programfejlesztés céljához fontos paramétereit a mellékletek között található a 12. táblázat tartalmazza. A táblázat sorait a CUDA környezettel kapott „deviceQuery” parancs kimenete adja. Az általam használt Nvidia CUDA API-t kezelő függvénykönyvtár verziószáma 11,8.

3.2.1. Numerikus eljárás kiválasztása

Az $1D$-s lehűlési görbék számítását a (2).-(4). egyenletek felhasználásával valósítottam meg, először CPU-ra, majd a CPU algoritmust átültettem GPU-ra, végül az így kapott GPU kódot optimalizáltam a választott kártyára. Akkor tekintettem jónak a GPU implementációt, amikor a GPU kód ugyanazt az eredményt adta, mint a CPU implementáció. Az alkalmazott numerikus eljárás választásához a sebességet és a pontosságot is vizsgáltam. A vizsgálatba bevont módszerek: FDM implicit és explicit [67–70], Crank-Nicolson [71], Runge-Kutta 4^{th} [72], Euler implicit és explicit eljárások [73].

A fenti eljárások közül személyes beszélgetések alapján és a választott GPU kártya korlátai miatt az explicit FDM eljárást választottam a lehűlési görbék számításához, ami sebességben, pontosságban és bonyolultságban is – egyelőre – egy kompromisszumos megoldást jelent az összes többi vizsgált algoritmus között. Az implementált algoritmus helyességének ellenőrzéséhez az implicit Euler eljárást vettem igénybe.

3.2.2. A választott numerikus eljárás GPU implementációja

A vizsgálatok során figyelembe kellett vennem, hogy a GPU kártya shared memória mérete korlátos, vagyis a lehűlési görbe és annak számításához szükséges minden adat nem foglalhat akármekkora tárterületet, azért hogy minden adat elférjen benne ahhoz, hogy ne a GPU lassú globális memóriát kelljen használni. Ez abból a szempontból érdekes, hogy a munkadarab sugár (r) irányát hány részre osztjuk (N, a grid pontjainak a száma), vagyis a lehűlési görbe milyen pontosságú lesz. Az egyszeres pontosságú lebegő pontos számok esetén egy szám 4 byte-ot foglal el a memóriából. Ebből következik, hogy ha a shared memória esetemben 48 Kbyte blokkonként, akkor a shared memóriába 12288 darab egyszeres pontosságú lebegő pontos számot lehet maximálisan tárolni. Emiatt nem lehet akármilyen finom felbontást alkalmazni sugár irányú hőmérsékletek tárolására. Ráadásul a 7-től kezdődő compute capability-vel rendelkező kártyákban a nem-lefoglalt shared memória jelenti az L1 cache tárhelyét is, aminek a szükségesnél nagyobb mértékű csökkenése negatívan hat a futó GPU program sebességére is. Viszont ha az adatok egy részét tárolom csak a shared memóriában, akkor a lassú globális memória olvasgatása miatt a GPU program végrehajtás ideje megnő, valamint a globális memória olvasási korlátai miatt divergencia lép fel a futó szálak között, ami még tovább lassítja a program futását. A GPU kártya szálkezelésének köszönhetően csak 32-vel osztható grid méretet javasolt választani, ami tovább szűkíti a választható grid méretét. Emiatt – esetemben – a szóba jöhető grid méretének 32-t, 64-t, 96-t, esetleg 128-t érdemes választani, a CPU esetében teljesen szabadon választott értékkel szemben. Minél nagyobb grid méretet választok, annál kevesebb lehűlési görbét lehetséges egyidejűleg kiszámolni.

A CPU-ban implementált és sebesség optimalizált algoritmus egy az egyben átemelése a GPU-ra sajnos nagyon gyatra futási időket produkált, mivel a GPU speciális futási környezete nagyon eltér a CPU esetén megszokottól. A teljesség igénye nélkül a GPU kártya programozásakor figyelembe kellett vennem a következőket:

- A grid méretének 32-vel maradék nélkül osztható értéket javasolt választani.
- A számításokhoz szükséges adatokat a shared memóriába érdemes tárolni.
- A szálak egy warp-on belüli divergenciája kerülendő.
- A lehető legkevesebb shared memória használata.

- A sokszor használt változók a shared memóriában foglaljanak helyet.
- A lehető legkevesebb művelet alkalmazása az osztás műveletek kerülésével.
- A globális memória használata kerülendő. Viszont amikor ez elkerülhetetlen, akkor a „coalesced" memóriaelérés használata javasolt.
- Törekedni kell, hogy a cuda occupancy a lehető legnagyobb legyen. A nagyobb occupancy nem jelent azonnal gyorsabb programot, csak nagyobb – elvileg elérhető – „hatékonyságot".
- A lehető legtöbb változó tárolása a regiszterekben, viszont a túl sok regiszter használata csökkenti az egy multiprocesszoron egyszerre futtatható szálak számát, vagyis egyben csökkentheti a program futásának a sebességét azáltal, hogy kevesebb lehűlési görbét képes egyszerre kiszámolni,

Látható, hogy a fenti megfontolások sokszor egymásnak ellentmondó korlátokat támasztanak a programmal szemben, és mindegyiket figyelembe véve, csak bizonyos kompromisszumok árán lehetséges azok alkalmazása. Ennek fényében az általam kidolgozott adatpárhuzamos végrehajtási modell a következő tulajdonságokkal bír:

- 32 bites egyszeres pontosságú lebegő pontos számokkal dolgoztam.
- A szükséges szál local változókon kívül minden nem tömb változó a statikus shared memóriában foglalt helyet.
- A dinamikus shared memóriát használtam a bementi HTC idő- és hőmérsékletpontjainak, a számolt lehűlési görbe idő- és hőmérsékletpontjainak, valamint a referencia lehűlési görbe idő- és hőmérsékletpontjainak a tárolására, valamint a számításokhoz szükséges HTC függvény lineáris interpolációval előreszámolt értékeinek tárolására.
- A grid 0-dik és utolsó pontjának a számítását külön warp-ban számoltam, a divergencia elkerülése miatt.
- A szükséges shared memóriának a méretét dinamikusan számoltam a grid méretéből úgy, hogy az interpolációs HTC függvény értékek számára minimum 32 érték álljon rendelkezésre annak érdekében, hogy a legkevesebb shared memóriát használhassam.
- A konfigurációs paraméterek a konstans memóriát foglalják.
- A bemeneti paraméterek shared memóriába másolását a rendelkezésre álló összes szál párhuzamosan másolja, figyelve a „coalesced" memóriaelérésre.
- Minden időtől és hőmérséklettől nem függő értéket a GPU kód meghívása előtt kiszámoltam.
- Egy blokk egy lehűlési görbe számolásáért felel.
- Figyelembe kellett venni a számolt lehűlési görbe pontosságát abban a tekintetben, hogy a **Shannon-Nyquist-féle mintavételezési tétellel** [74, 75] összhangban rögzítsem a számolt

eredményeket. Különben az ebből fakadó apró pontatlanságok és az egyszeres pontosságú lebegőpontos számok kényszerű használata miatt a két pontatlansági faktor együtt számottevő számítási hibához vezetne.

A számításokhoz szükséges adatoknak az alapgép memóriájában előre foglaltam nem lapozható memóriát annak érdekében, hogy a háttérben történő kétszeres adatmásolást elkerüljem. Erre azért volt szükség, mert ha az adatok az alapgépen lapozható memóriaterületre kerülnek (ez az alapértelmezett beállítás), akkor CPU és GPU memóriája közötti adatmásolást végző függvények a háttérben először átmásolják az adatokat egy nem lapozható memóriaterületre, majd csak ezután másolódnak az adatok a GPU globális memóriaterületére. A tényleges másolás után, amikor a HTC görbék pontjait a GPU globális memóriájába másoltam, a szükséges shared memória méretének kiszámolását végeztem el, majd meghívtam a GPU kernel-t a szükséges blokkok, a blokkokban futtatott szálak, és a shared memória méretének, valamint a szükséges adatok memóriacímének átadásával.

A GPU program első részében az előre számolható paraméterek meghatározása után minden blokk számára meghatároztam a shared memóriában az adott blokknak szóló adatok pontos címét, és átmásoltam a globális memóriából a shared memória adott címére az adatokat. Amennyiben a GPU dinamikus shared memóriát használ, akkor egyetlen címmel lehet a teljes lefoglalt shared memóriát címezni, ami miatt némi pointer-aritmetikára van szükség. Könnyebbség, hogy minden blokkhoz tartozó shared memória ugyanazon a címterületen található, így itt nincs szükség még egy címtranszformációra. Mivel ebben az algoritmusban egy blokk számol ki egy lehűlési görbét és egy blokk szálai a hőmérsékletet a munkadarab sugarában, viszonylag könnyű dolgom volt. Az adatmásoláskor kihasználtam, hogy 1 blokknyi szál áll rendelkezésre az összes adat másolásához, tehát a szálak számával egyenlő adatot egyszerre tudtam másolni a globális és a shared memória között. Az adatmásolás után beállítottam az algoritmus futásához szükséges kezdeti hőmérsékleteket és egyéb paramétereket, majd elindítottam a lehűlési görbét meghatározó ciklust. Az algoritmusnak elméletileg 4 időtengelyt kell szinkronban tartania: számolt és mért lehűlési görbe időtengelyeit, a HTC függvény időtengelyét valamint az algoritmus maximális időlépéséből számolt időtengelyt, ami várhatóan jelentősen sűrűbb mint a másik három. Az összes időtengelyt a mért lehűlési görbe tengelyéhez kell igazítani, miközben az algoritmus időtengelye nem választható meg teljesen szabadon, itt az időlépés nem lehet egy maximális értéknél nagyobb. Ezen felül az időtengelyekről (kivéve az algoritmus időtengelyét) nem lehet általánosan kijelenteni, hogy egyenletes elosztásúak. Az időtengelyek összehangolása miatt a HTC göbe adatait lineáris interpoláció segítségével áttranszformálom a mért lehűlési görbe időtengelyére. A számolt lehűlési görbe időtengelye meg fog egyezni az algoritmus

által számolt lehűlési görbe időtengelyének megfelelő részhalmazával, amit azzal érek el, hogy a mért lehűlési görbe időtengelyének pontjai előtt és után számolt lehűlési görbe időtengelyének pontjaiból képzett függvény megfelelő pontjaiban lineáris interpolációt hajtok végre. Ennélfogva az algoritmus időlépésének nagyságával már nem kell foglalkozni a feltétlenül szükséges mértékben jobban. Amint megvan minden szükséges idő- és hőmérsékletpont a számolt lehűlési görbéhez, visszamásoltam a hőmérsékletértékeket a GPU shared memóriából a GPU globális memóriába, valamint egy újabb lépésben a GPU globális memóriájából az alapgép memóriájába. Az algoritmus működését az 1. algoritmus mutatja be.

A pszeudokód 3 eljárásból áll. Sorrendben az utolsó eljárás (ami a CPU-n fut) előkészíti a GPU kód futásához szükséges paramétereket, memóriafoglalásokon keresztül az adatmásolásokig, elindítja a gpu kódot, valamint az eredményeket visszamásolja a host gép memóriájába. A középső eljárás végzi a számításhoz szükséges adattranszformációk elvégzését mint például a HTC görbe interpolációs számításait, vagy a kezdeti értékek meghatározását. Ezután elindítja az algoritmus időtengelyének bejárását, miközben csak a mért lehűlési görbe időtengelyének pontjaiban tárolja el hőmérsékletértékeket. Ezt úgy éri el, hogy amikor a következő időlépésben az algoritmus ideje nagyobb lenne mint a soron következő mért lehűlés görbe időtengelyének értéke, lecsökkenti az időlépést pontosan akkorára, hogy a következő lépéssel a megfelelő időpontba számolja ki a hőmérsékletértékeket, majd a számítások után az időlépés értékét visszaállítja az eredeti értékre. A ciklus végeztével meghatározza a célfüggvény értékét, valamint a shared memóriából átmásolja a globális memóriába a számolt hőmérsékletértékeket. A . . . _IN_PARALLEL végű másoló és transzformációs függvények az adatokon végzendő műveleteket, az egy GPU-blokkban definiált szálak számához igazítva párhuzamosan végzi. A CALULATE_TEMPERATURES() eljárás számolja a sugár irányú hőmérsékletértékeket, minden szál egy-egy pontját szintén párhuzamosan.

A számítások végén a 3.3. fejezetben ismertetett eredményeket kaptam. Az elkészült adatpárhuzamos modell fejlesztése közben aktívan használtam a CUDA Occupancy Calculator-t a helyes konfigurációk megválasztásához, a CUDA nsight-compute éppen aktuális verzióját, hogy kiderüljenek az implementáció korlátai, és azok lehetséges okai.

3.3. Eredmények értékelése

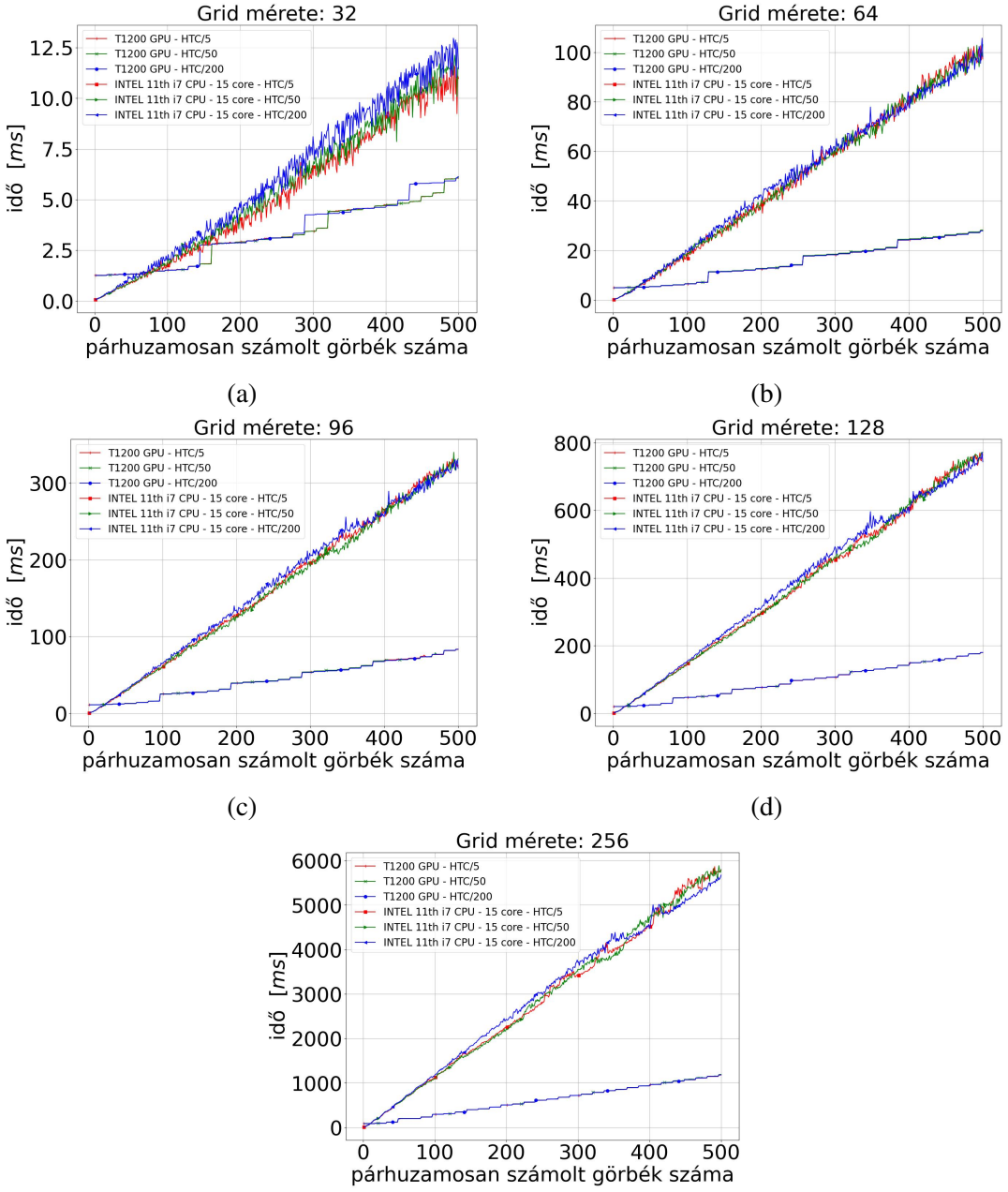
A CPU-ra és GPU-ra fejlesztett algoritmusok sebességét teszteltem 32, 64, 96, 128 és 256 grid méret és 5, 50 és 200 pontból álló HTC görbe esetére is. Mindezt annak tudatában, hogy matematikai szempontból a lehűlés görbéket számoló eljárások sebességére nem szabad hogy jelentős hatással legyen a HTC görbe pontjainak a száma. A grid mérete viszont annál inkább befolyásolja a számítási

1. Algoritmus Lehűlési görbe egyszerűsített számítása GPU-ban

```
1  procedure CALCULATE_TEMPERATURES(temperatures, htc_data)
2      if threadIdx.x = 0 then    ▷ középvonal
3          alfa ← CALCULATE_ALFA(temperatures.prev[ 0 ])
4          temperatures.new[ 0 ] ← temperatures.prev[ 0 ] + alfa · ( temperatures.prev[ 1 ] - temperatures.prev[ 0 ] )
5      end if
6      if threadIdx.x > warp_size  and  (threadIdx.x < last_temp_index + warp_size) then    ▷ közbenső pontok
7          idx ← threadIdx.x - warp_size;
8          alfa ← CALCULATE_ALFA(temperatures.prev[ idx ])
9          temperatures.new[ idx ] ← temperatures.prev[ idx ] +
                   + alfa · ( c1 · temperatures.prev[ i + 1 ] + c2 · temperatures.prev[ i - 1 ] - 2 · temperatures.prev[ idx ] )
10     end if
11     if threadIdx.x = last_temp_index + warp_size then    ▷ felület
12         k ← CALCULATE_K(temperatures.prev[ last_temp_index ])
13         alfa ← CALCULATE_ALFA(temperatures.prev[ last_temp_index ])
14         temperatures.new[last_temp_index] ← temperatures.prev[ last_temp_index ] +
                   + alfa · ( temperatures.prev[ last_temp_index - 1 ] - temperatures.prev[ last_temp_index ] ) -
                   - htc_data · k · ( temperatures.prev[ last_temp_index ] - end_temp )
15     end if
16 end procedure
17
18 procedure CALCULATE_COOLIG_CURVES_IN_GPU(allocated_gpu_memory_pointers: reference_times, htc_data, cc_data)
19     if blockIdx.x < cc_num then
20         interpolated_htc_data ← HTC_LINEAR_INTERPOLATION_IN_PARALLEL(reference_times, htc_data)    ▷ a HTC függvény pontjainak lineáris
           interpolációja a mért lehülési görbe időadataira
21         COPY_GLOBAL_DATA_TO_SHARED_MEMORY_IN_PARALLEL(reference_times, interpolated_htc_data, cc_data)
22         FILL_DATA_IN_PARALLEL(temperatures.prev, start_temp)    ▷ kezdeti hőmérséklet beállítása
23         time ← 0
24         STORE_CC_DATA(time, temperatures)    ▷ kezdeti hőmérséklet tárolása
25         time ← delta_t
26         reference_time ← GET_NEXT_REFERENCE_TIME(time, reference_times)    ▷ a mért lehűlési görbe következő időadata
27         while time < end_time do
28             CALCULATE_TEMPERATURES(time, temperatures, interpolated_htc_data)
29             if time > reference_time then    ▷ a hőmérséklet tárolásának ideje
30                 reftime_temperature ← TEMPERATURE_LINEAR_INTERPOLATION(prev_time, time, prev_temperature, temperatures.new, referen-
                   ce_time)
31                 STORE_CC_DATA(reference_time, reftime_temperature)
32                 reference_time ← GET_NEXT_REFERENCE_TIME(time, reference_times)
33             end if
34             time ← time + delta_t
35             SWAP(temperatures.prev, temperatures.new)    ▷ az előző és az új hőmérsékletadatok cseréje
36         end while
37         COPY_SHARED_DATA_TO_GLOBAL_MEMORY_IN_PARALLEL(stored_temperatures)
38     end if
39 end procedure
40
41 procedure GPU_COOLING_CURVE_CALCULATION(config_data, cc_data, htc_data)
42     CALCULATE_ALL_NECESSARY_CONSTANT_PARAMETER()
43     allocated_gpu_memory_pointers ← ALLOCATE_PINNED_GPU_MEMORY(config_length, cc_length, htc_length, temperature_length)
44     COPY_DATA_HOST_TO_GPU(allocated_gpu_memory_pointers, config, reference_times, htc_data, cc_data)
45     shared_mem_size ← CALCULATE_GPU_SHARED_MEMORY_SIZE(cc_length, htc_length, temperature_length)
46     thread_num ← grid_length + 2 · warp_size
47     block_num ← cc_num
48     CALCULATE_COOLIG_CURVES_IN_GPU«BLOCK_NUM, THREAD_NUM, SHARED_MEM_SIZE»(allocated_gpu_memory_pointers)    ▷ a GPU kód
       indítása
49     COPY_DATA_GPU_TO_HOST(stored_temperatures, objective_function)
50     DEALLOCATE_GPU_MEMORY(allocated_gpu_memory_pointers)
51 end procedure
52
53 Előre kiszámolt állandó paraméterek: delta_t, c1, c2, alfa és k változókba épített egyéb előre számolható paraméterek
54 Konfigurációs paraméterek: warp_size, grid_length, cc_num, end_time, last_temp_index, start_temp, end_temp
55 Parallel típusú eljárások szálainak száma ( . . . _IN_PARALLEL() függvények esetében ): thread_num
```

sebességet. Az eredményeket a 10. ábra mutatja be. Az ábrákról leolvasható, hogy a GPU számítási sebessége nagyon stabilnak tekinthető, míg a 32-es grid méret esetében van kis eltérés a 200 pontból álló

HTC görbe számításakor, valamint az algoritmusok lényegében függetlenek a HTC görbe pontjainak a számától. Furcsa módon a CPU is és a GPU is gyorsabban határozza meg a lehűlési görbét kevesebb pontból álló HTC görbék számításának esetében. Ez az apró eltérés jól mutatja, hogy van még lehetőség az algoritmusok további optimalizációjára.



10. ábra. Lehűlési görbék számítási sebessége különböző grid méretek és CPU/GPU esetében

A 10. ábrák bal oldalán egy bizonyos párhuzamosan számolt görbe esetében a GPU architekturális okok miatt lassabb mint a CPU, ugyanis egy kernel (a GPU program) indítása a GPU kártyán mindig több időbe kerül, mint a CPU-k esetében egy függvény meghívása, többek között azért is, mert a

számításokhoz szükséges adatokat el kell juttatni a GPU kártyára, ami kártyától, bemeneti adatoktól, felhasznált memóriától, stb. függően lassabb indulást eredményez, mintha csak CPU-t használnánk. Ezt az időveszteséget a párhuzamosan számolandó görbéktől függően a GPU ledolgozza.

A számítási eredmények alapján a grid méretének 96-ot választottam, annak érdekében, hogy a lehűlési görbék felbontásából adódó pontatlanság ne vigye félre a tesztelt keresőalgoritmusok futását, valamint 3.000.000 lehűlési görbe GPU-val történő kiszámítása 10 percen belül befejeződik (a választott GPU kártyán), ellentétben a 128-as vagy 256-os grid méretek esetében. (Arra számítok, hogy az 5. és a 6. fejezetekben ismertetett eljárások maximum 3.000.000 lehűlési görbe kiszámolása esetén már képesek lesznek elérni egy egyfajta optimumot.) Emellett korlátoztam a regiszterek számát 32-re a nsight-compute ajánlása alapján. A GPU teljesítményét legjobban az 5.8. és a 6.9. fejezetekben ismertetett M2., M3. és a 7. táblázatok tartalmazzák. A CPU és a GPU futási eredményeinek összehasonlításához az M1. táblázat tartalmazza a CPU-val mért futási eredményeket.

*3.4. **1. Tézis***

Kidolgoztam egy új típusú, GPU-ra optimalizált adatpárhuzamos számítási eljárást az 1D tengely-szimmetrikus (hengeres) végtelen hosszú test hőmérséklet-eloszlásának hatékony számítására, amely a számítási eljárás során figyelembe veszi a CPU és GPU architekturális lehetőségeit és korlátainak megfelelő optimalizációs eljárásokat alkalmaz a maximálisan elérhető számítási sebesség, valamit a párhuzamosan számolható lehűlési görbék számának növelése érdekében. Validációs tesztek eredményeivel igazoltam, hogy az új típusú számítási megoldás – azonos pontosság mellett – jelentősen gyorsabb a CPU-ra készült ismert számítási módszereknél.

3.5. Tézishez kapcsolódó saját publikációk

1. Sandor Szenasi, Zoltan Fried és Imre Felde: „GPU Accelerated Heat Transfer Simulation Supporting Heuristics To Solve The Inverse Heat Conduction Problem”. *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE, 2020. jan., 287–292. old. DOI: 10.1109/sami48414.2020.9108768.

4. Inverz Hőtani Probléma Közelítő Megoldása Neurális Hálózattal

Az 5.-7. fejezetekben részletesen ismertetek inverz hőtani probléma megoldására különféle módszereken alapuló eljárásokat. A probléma jellemzőire való tekintettel (rosszul feltett matematikai probléma, nagy számításigény, stb.) a megoldások jelentős része valamilyen heurisztikus keresésen alapul, mint például a különféle gradiens alapú, vagy populáció alapú módszerek. Ezek a módszerek a gyakorlatban jól használhatóak, azonban számos probléma [76] merül fel az implementációjuk során.

A gépi tanulási technikák már számos helyen megjelentek és használatosak az inverz műszaki számítások területén. Az alapötlet már meglehetősen korán előkerült [77, 78], azonban a kezdeti időszakban még nem álltak rendelkezésre a megfelelő számítási teljesítményt nyújtó számítógépek. A módszer előnye azonban már ekkor is nyilvánvaló volt, hiszen ennek segítségével egy nem-iteratív megoldást lehetne találni a már ismert, nehezen megoldható problémára [79, 80].

A témához kapcsolódó kutatások tehát alapvetően sikeresek voltak, azonban az így nyert megoldások számos korlátot is tartalmaznak. Legnagyobb korlátnak a pontosság kérdése tekinthető, hiszen egy neurális hálózatokon alapuló modell várhatóan nem fogja tudni megtalálni a keresett hőátadási tényező pontos értékét. Ezért ezen módszerek önálló direkt alkalmazása nem kielégítő, érdemes azonban megjegyezni, hogy jelen esetben nincs is erre szükség. Ha a neurális hálózat egy megfelelő közelítő értéket tud adni, az már segítséget jelent az erre épülő további heurisztikus módszereknek, hogy ez alapján végezzék el a kezdő (most már csak részben) véletlen inicializálást. Szintén a témához kapcsolódó motivációmként megjegyezhető, hogy a neurális hálózatokkal kapcsolatos módszerek nagy fejlődésen mentek keresztül az utóbbi években.

A témával foglalkozó irodalom jelentős része a napjainkban tapasztalható újabb gépi tanulási „aranykor” előtt jelent meg. Mostanában azonban már jóval erősebb hardver és kiforrottabb gépi tanulási keretrendszerek érhetőek el, így érdemes újra feleleveníteni a témát. Ezért ebben a fejezeteben egy gépi tanulási modellt mutatok be, ami az 1D tengelyszimmetrikus test felületén kialakult hőmérsékleti görbe alapján a felületi HTC függvény becslésére alkalmas.

A heurisztikus keresésen alapuló módszerek a gyakorlatban jól használhatóak, azonban számos probléma merül fel az implementációjuk során:

– A heurisztikus módszerek első lépése tipikusan valamilyen véletlen inicializálás szokott lenni. Ez az inicializálás pedig gyakran nagy hatással van az algoritmus későbbi futására is. Amennyiben a populáció elemei közül legalább néhány közel indul a leendő megoldáshoz, akkor várhatóan ezek

gyorsan és nagy bizonyossággal meg is fogják azt találni. Amennyiben azonban ilyen elemek nincsenek, akkor a keresés kimenete jóval bizonytalanabb, és nagy valószínűséggel tovább is fog tartani.

– Szintén ideálisnak tekinthető, ha az induló elemek lefedik a teljes keresési teret. A populációalapú módszerek esetében általában a véletlen inicializálás során több száz, vagy akár több ezer elem kezdőpozícióját határozzuk meg. Amennyiben ezek a keresési tér távoli részeiben helyezkednek el, akkor nagyobb eséllyel jól feltérképezik azt, így várhatóan megtalálják a globális optimumot. Ha azonban a véletlen inicializálás egymáshoz közeli helyekre teszi őket, akkor elképzelhető, hogy az elemek csak azt a kisebb régiót fogják tudni átvizsgálni (főleg az olyan jellegű módszereknél, mint például a genetikus algoritmus, ahol az elemek főleg a már meglévő paramétereket cserélgetik egymás között).
– Harmadik problémaként pedig a lokális minimumba való befutás valószínűségét érdemes megemlíteni. Az inverz hőtani probléma esetén gyakran tapasztalható, hogy egymástól jelentősen eltérő formájú hőátadási függvények is képesek hasonló lehűlési görbéket eredményezni. Az, hogy maga a módszer ezek közül melyiket találja meg, nagyban függhet attól, hogy milyen kezdőállapotokból indítottuk el magát a heurisztikus keresést. Szerencsétlen állapotból indított keresések gyakran adnak jó célfüggvényértéket eredményező, de valójában mégis elfogadhatatlan alakú függvényeket.

A fentiek alapján látható, hogy a heurisztikus módszerek teljesen véletlenszerű indítása számos problémával jár. Segítséget jelentene, ha legalább egy közelítő megoldással rendelkeznénk a megoldandó feladathoz. Ilyenkor lehetővé válna, hogy a véletlen indítást ez alapján vezéreljük, és a kezdő populáció elemeit ezen kezdőmegoldás valamilyen környezetében helyezzük el.

Ezt a közelítő megoldást lehetőség szerint valamilyen egyszerű, gyorsan végrehajtható módszerrel kell megtalálni (hiszen egy újszerű heurisztika indítása ismét felvetné az előzőleg már említett problémákat). A gépi tanulás területén már meglévő módszerek alkalmasnak tűnnek egy ilyen jellegű előfeldolgozáshoz. Egy megfelelően betanított neurális hálózat képes lehet arra, hogy egy megadott lehűlési függvényhez meghatározza az azt okozó hőátadási tényező függvényt.

A neurális hálózatok tanítása során mindig kritikus fontosságú a megfelelő mennyiségű és minőségű tanítási minta megléte. Az általam kifejlesztett GPU alapú szimulációk segítségével elfogadható idő alatt van lehetőség nagy mennyiségű szimuláció lefuttatására, így a tanításhoz szükséges hőátadási függvény – lehűlési görbepár előállítására.

4.1. A modell paramétereinek meghatározása

A felügyelt tanulás alapelve, hogy a rendszernek szüksége van egy úgynevezett tanító mintára, ami tartalmazza a bemeneti, és az azokhoz tartozó kimeneti adatokat. Jelen esetben a neurális hálózat célja a lehűlési adatok alapján meghatározni az azokhoz vezető hővezetési tényezőt. A modell során az alábbi paramétereket használtam:

- Feladat jellege: 1 dimenziós hőátadás (végtelen hosszúságú test modellezése).
- Kezdőhőmérséklet: $850°C$.
- Hűtőközeg hőmérséklete: $20°C$.
- Munkadarab sugara: $6{,}25mm$.
- Kísérlet hossza: $60sec$.

A hőmérsékleti adatok $0{,}5sec$ mintavételezéssel készültek, tehát a bemenet összesen 121 darab lebegőpontos számot tartalmaz. A HTC függvény leírásához $0°C$ és $850°C$ közötti $10°C$-os felbontást használtam, tehát a kimenet összesen 86 darab lebegőpontos értéket tartalmaz.

4.1.1. Hőátadási tényező leírása

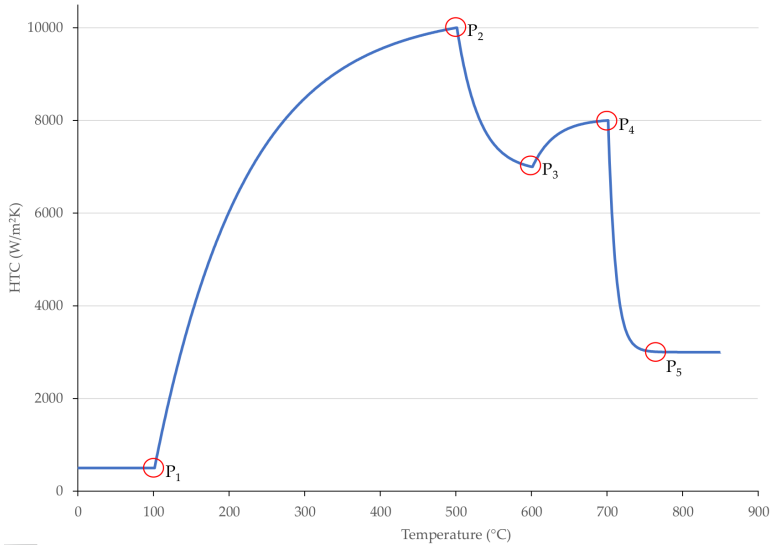
A feladat megoldásához szükséges megfelelő számú hőátadási tényező függvény generálása. Ehhez kidolgozásra került egy újszerű módszer, amely során vezérlőpontok segítségével le lehet írni az egyes függvények karakterisztikáját, ez alapján pedig tetszőleges felbontással le lehet generálni magukat a függvényeket is. A módszer alapja az, hogy van $N_v \in \{1, 2, 3 \ldots\}$ darab vezérlőpontunk $(P_1, P_2, \ldots, P_{N_v})$. Mindegyik pont úgy képzelhető el, hogy van egy HTC („htc") illetve hőmérséklet („temp") komponense, amelyből a vízszintes „temp" koordináta mutatja a ponthoz tartozó hőmérsékletet, függőleges „htc" koordináta pedig a hozzátartozó hőátadási tényező értéket. Ezek a pontok fogják majd megmutatni a generálandó függvényben megtalálható töréspontokat (11. ábra).

A vezérlőpontok alapvetően bármilyen HTC és hőmérséklet értékeket felvehetnek, amennyiben azok megfelelnek az alábbi megszorításoknak. A hőmérséklet érték megszorítását a (10) egyenlet

$$0°C \leqslant P_{i,temp} \leqslant 850°C, \quad \forall i \in \{1,2,\ldots,N_v\} \tag{10}$$

a hőátadási tényező érték megszorítását pedig a (11) egyenlet fogalmazza meg:

$$0\frac{W}{m^2 \cdot K} \leqslant P_{i,htc} \leqslant 12000\frac{W}{m^2 \cdot K}, \quad \forall i \in \{1,2,\ldots,N_v\} \tag{11}$$



11. ábra. Hőátadási tényező generálása során használt vezérlőpontok

Sorrendbeli megszorításként a $P_{1,temp} < P_{2,temp} < \ldots < P_{N_v,temp}$ feltételeket alkalmaztam. Hogy a függvény folytonos legyen a kitűzött $0°C$ és $850°C$ tartományban, mindkét oldalról ki kell egészíteni egy-egy virtuális ponttal (P_0 és P_{N_v+1}, ahol $P_{0,temp} = 0°C$, $P_{0,htc} = P_{1,htc}$, valamint $P_{N_v+1,temp} = 850°C$ és $P_{N_v+1,htc} = P_{N_v,htc}$.

Külön figyelmet érdemel az egyes pontokat összekötő görbe jellemzése, hiszen a 11. ábrán is jól látható, hogy ez általában nem egyenes, annak alakja ettől jelentősen eltérhet. Ezért bevezettem minden pont esetén egy speciális (α_i) értékeket, amelyek meghatározzák az egyes pontokat összekötő görbék formáját: $-1{,}0 \leqslant \alpha_i \leqslant +1{,}0, \quad \forall i \in \{1, 2, \ldots, N_v\}$

Egy megadott T hőmérsékletű helyen a hőátadási tényező értéke tehát kiszámítható a vele szomszédos P_i és P_{i+1} pontok és az első ponthoz tartozó α_i értéke alapján:

$$\alpha_i' = \frac{1}{\alpha_i C_\alpha}, \qquad \delta(T) = \frac{T - P_{i,temp}}{P_{i+1,temp} - P_{i,temp}}, \qquad \gamma(T) = \frac{1 - e^{\frac{-\delta(T)}{\alpha_i'}}}{1 - e^{-\frac{1}{\alpha_i'}}} \tag{12a}$$

$$h(T) = \gamma(T) \cdot (P_{i+1,htc} - P_{i,htc}) \tag{13}$$

Ahol $h(T)$ a T hőmérséklethez tartozó hőátadási tényező értéke, a C_α pedig egy skálázáshoz használható konstans paraméter. Amennyiben az α_i paraméter értéke 0, akkor az egy nullával való osztáshoz vezetne, ezért ilyenkor az alábbi lineáris átmenetet kell használni:

$$h(T) = \delta(T) \cdot (P_{i+1,htc} - P_{i,htc}) \tag{14}$$

4.1.2. Potenciális hőátadási függvények generálása

Az előzetes vizsgálatok alapján nyilvánvaló, hogy a feladat meglehetősen komplex, hiszen a neurális hálózatnak nagy mennyiségű adat (4.1.3.-4.1.4. fejezetek) között kell nemlineáris összefüggéseket találnia.

	Hőmérséklet ($°C$)		Hőátadási tényező ($\frac{W}{m^2 \cdot K}$)	
	Min	Max	Min	Max
P_1	200	400	200	500
P_2	401	650	2000	12000
P_3	651	750	200	500
P_4	751	820	500	800
P_5	821	850	100	400

1. táblázat. A hőátadási tényező függvény generálása során használt korlátok az egyes vezérlőpontokra vonatkoztatva

Ennek megfelelően a várható, hogy a rendszer csak nagyméretű tanítási minta esetén képes megtanulni a szükséges mintázatokat. Ennek mérete előre nem jósolható meg, de várhatóan több milliós nagyságrendről beszélhetünk. Ezek alapján nyilvánvaló, hogy való világból származó adatok használata itt nem lenne kielégítő, mindenképpen generálnunk kell a szükséges tanítási adatbázist. A szakirodalomban fellelhető adatok alapján jól látható, hogy a hőátadási tényezőket leíró függvények alapvetően hasonló karakterisztikát követnek. Egy gyors felfutást követő tartományban egy lassabb csökkenés jelenik meg, esetenként egy újabb kisebb emelkedéssel, majd további csökkenéssel. Ezt felismerve, a generálást végző modellt úgy építettem fel, hogy a természetben megtalálhatóakhoz hasonló függvényeket generáljon, viszont mindezt úgy, hogy a potenciális függvények terét minél jobban leírják. Ennek megfelelően öt vezérlőpontra van szükség:

- P_1: az első emelkedő szakasz kezdete,
- P_2: az első csúcspont,
- P_3: a belső hullámvölgy alja,
- P_4: a második csúcspont,
- P_5: az utolsó csökkenő szakasz végpontja.

A vezérlőpontok HTC és hőmérséklet koordinátáit egy véletlen generálás határozza meg. Az egyes pontok lehetséges pozícióit a szakirodalomban fellelhető hőátadási tényező függvények vizsgálata alapján állapítottam meg (1. táblázat).

4.1.3. Neurális hálózat tanítási adatainak előállítása

A felügyelt tanuláshoz szükség van egy nagyméretű tanító adatbázisra, amelynek tartalmaznia kell a bemeneti és az azokhoz tartozó helyes kimeneti adatokat. Jelen esetben a kimenetet az előzőleg generált HTC görbék képviselik, a bemenet pedig az ezekhez tartozó lehűlési görbék sorozata lesz. Ehhez viszont elő kell állítani ezeket a lehűlési görbéket. Több millió görbéről és szimulációról van szó, tehát ez egy meglehetősen számításigényes feladat. Ehhez fel tudtam használni a már előzőleg elkészített GPU alapú hőátadási szimulációkat végző modult.

Az előzőekben elkészített véletlen hőátadási függvényeket eltároltam, amely a hőátadási függvény értékeket tartalmazta $0°C$ és $850°C$ között $10°C$-os lépésekben.

A GPU alapú hőátadási szimuláció modul egyesével betöltötte ezeket az elemeket, majd mindegyike lefuttatott egy szimulációt, valamint elvégeztem az adatok előfeldolgozását is, amit a következő fejezet tartalmaz, amely során előállt a neurális hálózat tanításához szükséges adatbázis.

4.1.4. Neurális hálózat tanítási adatainak előfeldolgozása

A tanításához szükséges bemeneti és kimeneti adatpárok (mindkét esetben egy-egy lebegőpontos számokat tartalmazó számsor) elkészítése után, nem célszerű az adatokat közvetlenül felhasználni a tanításhoz, hanem egy előzetes előfeldolgozást ajánlott azokon végrehajtani. Ennek lépései olyan függvénytranszformációk, amelyek segítségével a leendő modell várhatóan hatékonyabban fogja tudni megtanulni a két adatsor közötti kapcsolatokat. Így az előfeldolgozáshoz a következő 3 lépést alkalmaztam:

- **Invertálás**: A bemeneti és kimeneti függvények mértékegységei jelentősen különböznek egymástól. A bemenet hőmérséklet adatokat tartalmaz az idő függvényében, a kimenet pedig hőátadási tényező értékeket tartalmaz a hőmérséklet függvényében. Mivel mindkét függvényben megjelenik a hőmérséklet, célszerű ezeket hasonló mértékegységűre alakítani. Ennek érdekében invertáltam a bemenetet, vagyis a bemenet függvénye azt mutatja meg, hogy a munkadarab az egyes hőmérsékleteket ($0°C$ és $850°C$ között $10°C$ lépésközzel) melyik időpillanatban éri el (ez feltételezi, hogy a hőmérséklet monoton csökken). Ennek eredményeként mind a bemenet, mind pedig a kimenet a hőmérséklet függvényében írja le a megfigyelt jellemzőt. Ezzel a művelettel arra számítok, hogy könnyebb lesz köztük megtalálni az összefüggéseket.
- **Különbségek számítása**: Mielőtt az előzőleg előkészített függvényeket felhasználtam volna, megvizsgáltam a jelenség fizikai hátterét vagyis a HTC nagysága nem magára a tényleges hőmérsékleti értékre van hatással, hanem csak a hőmérséklet pillanatnyi változására. Mivel a

bemeneti adatok az idő függvényeként szerepelnek, ezért érdemes idősorként felhasználni a bementi adatsort. Viszont a lehűlési görbék nem stacionárius idősorok, vagyis a neurális hálózat felhasználásához megfelelő számú (esetemben egy) differenciaképzéssel stacionáriussá tehető. Ezért az előfeldolgozott (invertált) **lehűlés görbék függvényeiből** egy további transzformáció segítségével **előállítom azok változásfüggvényét**.

– **Normalizálás**: Ezt a lépést minden neurális hálózattal kapcsolatos feladat esetében célszerű megtenni. A hálózatok aktivációs függvényei általában 0 és 1, illetve −1 és 1 közötti értékeket adnak vissza. Bár a bemenetre nincs ilyen megszorítás, de tapasztalatból tudható, hogy az aktivációs függvények optimális módon többnyire a fenti intervallumok között használhatók, ennél nagyobb vagy kisebb értékek esetén a visszaadott értékek már kevésbé érzékenyek a bemenetre. Ennek megfelelően az előző lépésben nyert bemenő adatokat lineáris normalizálási [81] eljárással normalizáltam a

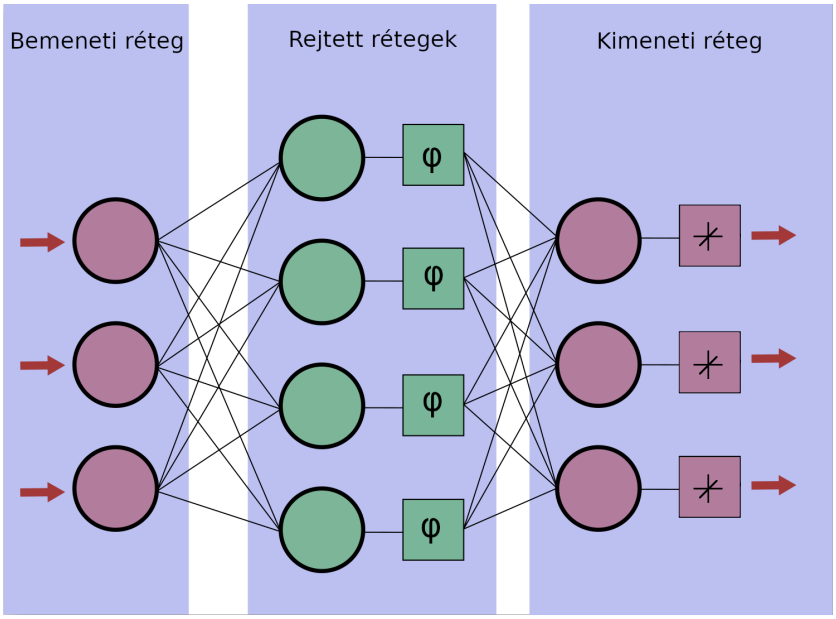
$$\mathrm{T_{norm}} = \frac{\mathrm{T} - \mathrm{T_{min}}}{\mathrm{T_{max}} - \mathrm{T_{min}}} \tag{15}$$

függvénnyel, amivel 0 és 1 közötti intervallumra transzformáltam az adatokat, ahol $T \in \mathbb{R}$ és $0°C \leq T \leq 850°C$. A fenti lépések segítségével előállt a neurális hálózat bemenetére kapcsolható végleges adatsor. A kimeneten várt HTC függvények esetében csak a lineáris normalizálást hajtottam végre, ahol $HTC \in \mathbb{R}$ és $200\frac{W}{m^2 \cdot K} \leq HTC \leq 12000\frac{W}{m^2 \cdot K}$.

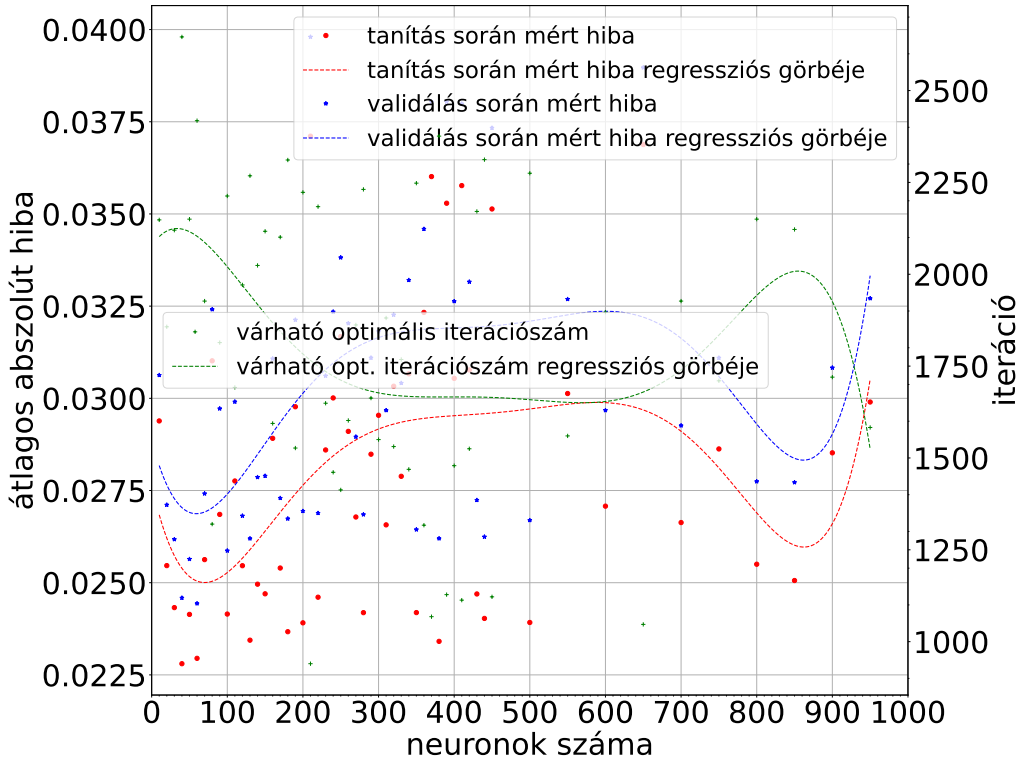
4.1.5. Neurális hálózat felépítése és tanítása

A neurális hálózatok képessége (és erőssége), hogy gyakorlatilag tetszőleges bemenet és kimenet között képesek megtalálni a leképezést elvégző függvényt (tehát ezek a hálózatok úgynevezett univerzális approximátorok). A neurális hálózatok általános felépítésére egy sematikus ábrát a 12. ábra mutat. Az általam kidolgozott neurális hálózat nagyon hasonló a 12. ábrához.

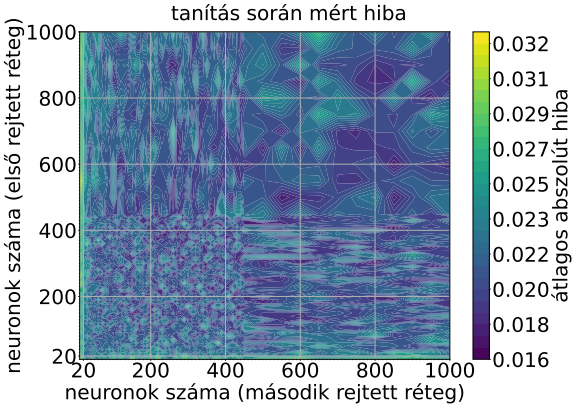
Annak érdekében, hogy el lehessen dönteni, hogy hány rejtett réteget kell alkalmazni a neurális hálózaton belül [82–84], illetve a rejtett rétegekben az optimális neuronok számát meghatározhassam, méréseket végeztem egy limitált adatbázison különböző neuronszámokkal. Ennek eredményeit egy rejtett réteg esetében a 13. ábra, két rejtett réteg esetében a 14. ábra mutatja:



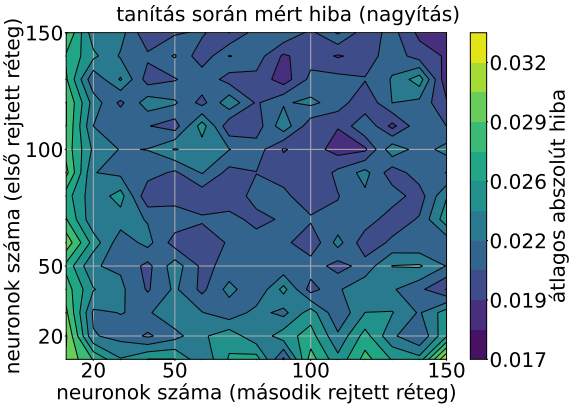
12. ábra. Neurális hálózat sematikus felépítése



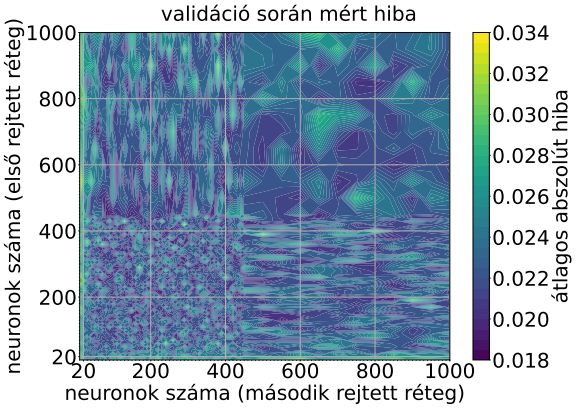
13. ábra. Egy rejtett rétegű hálózat rejtett rétegéhez tartozó neuron számának vizsgálata



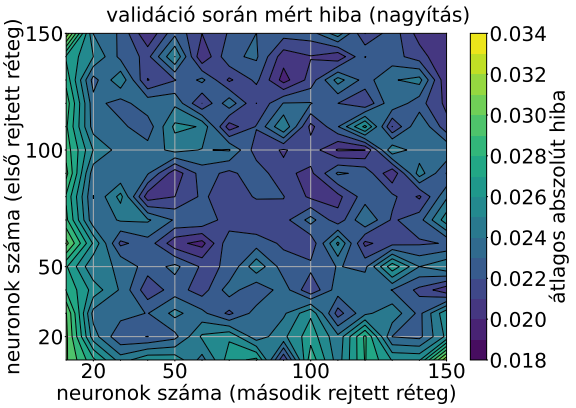
(a) 10 és 1000 neuron között



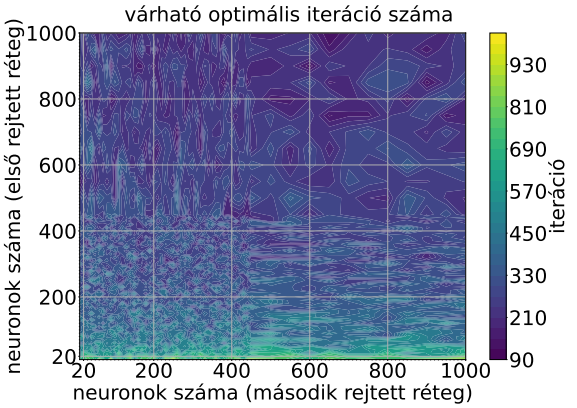
(b) 10 és 150 neuron között



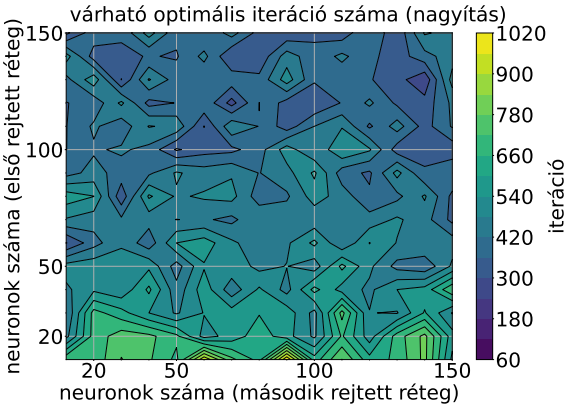
(c) 10 és 1000 neuron között



(d) 10 és 150 neuron között



(e) 10 és 1000 neuron között



(f) 10 és 150 neuron között

14. ábra. Két rejtett rétegű hálózat rejtett rétegeihez tartozó neuron számainak becslése

A 13. ábrán levő tanítás és validálás során mért veszteségfüggvények értékei eléggé szórnak, de az elmondható róluk, hogy 900 neuronig bármennyi neuron alkalmazásával el lehet érni a 0.04 átlagos abszolút hiba értéket, átlagosan 2000 iteráció után, a túltanulás elkerülése mellett. Azt az optimális iterációszámot, ami az alul- és a túltanuláshoz rendelt iterációszámok között található, a zöld keresztek jelzik az ábrán az egyes neuronszám függvényében [85]. Az optimális iterációszámot és az átlagos

abszolút hibát a neurális hálózat keretrendszerének segítségével határoztam meg, úgy, hogy a tanítási folyamatot úgy állítottam be, hogy a keretrendszer állítsa le a tanítás folyamatát abban az esetben, ha a validáláshoz tartozó átlagos abszolút hiba értéke 50 iteráción keresztül kisebb mint $1e^{-5}$, ami azt jelenti, hogy az optimális iterációszám a leállítás pillanatától visszafele számolt 50. iteráció.

Hasonló eredményeket tapasztaltam két rejtett rétegű hálózat esetében is, amelynek eredményeit a 14. ábra mutatja. Az a), c) és e) ábrák 10 és 1000 neuron között mutatják a mérések eredményeit különböző színek segítségével. A b), d) és f) ábrák a balra mellettük levő ábrák 10 és 150 neuron közötti terület nagyításai. Az a) és b) ábrák a tanítás során rögzített optimális átlagos abszolút hibát mutatja, a c) és d) ábrák a validálás során rögzített optimális átlagos abszolút hibát mutatja, az e) és f) ábrák pedig a tanítás során elért iterációs számot, amikor az optimális, validálás során mért átlagos abszolút hibát elérte a rendszer.

A 13. és a 14. ábrákat tanulmányozva az első rejtett réteg esetében 100 neuronban, a második rejtett réteg esetében pedig 120 neuronban határoztam meg a legkisebb neuronszámú rétegek számát, amivel a legalacsonyabb, validálás során rögzített optimális átlagos abszolút hiba elérhető a túltanulás elkerülése mellett. Így az alkalmazott hálózat alapvető jellemzői az alábbiak szerint alakultak:

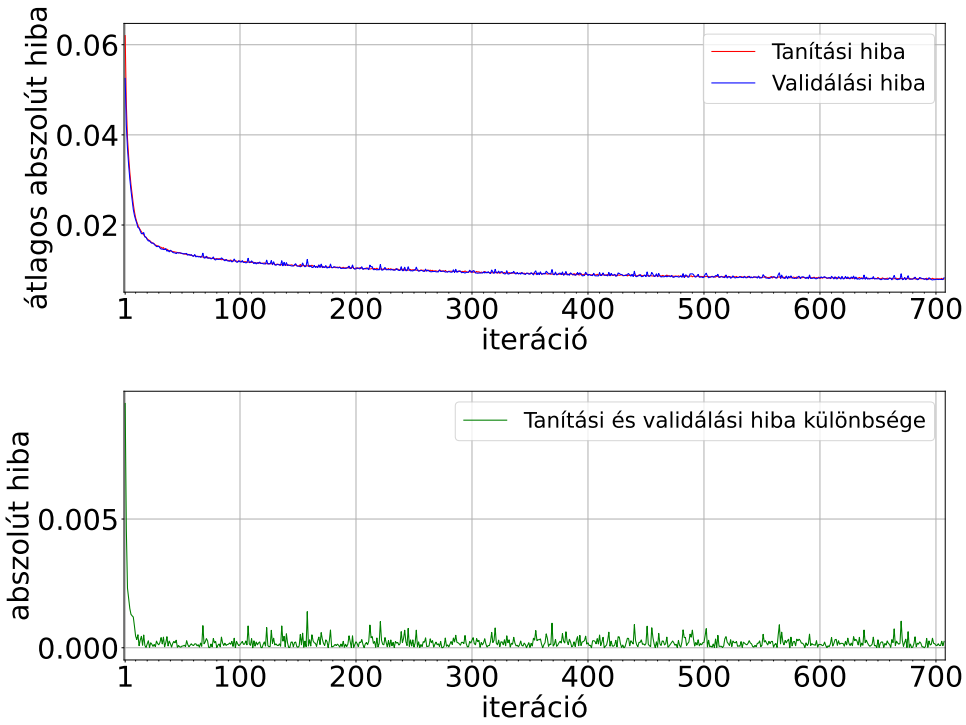
- 85 darab neuronból álló bemeneti réteg, ezek a neuronok fogadják a bemenő előkészített adatsort.
- 100 darab neuronból álló rejtett réteg lineáris aktivációs függvénnyel.
- 120 darab neuronból álló rejtett réteg szigmoid aktivációs függvénnyel.
- 86 darab neuronból álló kimeneti réteg lineáris aktivációs függvénnyel.
- Sűrű, előrecsatolt hálózat.
- ADAM optimalizációs eljárás.
- Átlagos abszolút hibafüggvény minimalizálása.

A bemeneti oldalon a 85 neuron a $10°C$ lépésközzel $0°C$ és $850°C$ között felvett lehűlési görbék hőmérsékleteinek különbségeiből következik, a kimeneti oldalon található 86 neuron a $10°C$ lépésközzel $0°C$ és $850°C$ között felvett HTC függvény hőmérsékleteinek számából származik. A hálózat a tanítás során az átlagos abszolút hiba hibafüggvényét próbálta minimalizálni. Az eredmények korrekt kiértékelése érdekében a tanítás során nem használtam fel az adatok 30%-át. A tanításhoz fel nem használt minták $2/3$-a a validációs adatbázist képezte, a maradék $1/3$-án teszteltem a tanított hálózatot. A teljes adatbázis egymillió HTC függvény–lehűlési görbe párt tartalmazott.

4.2. Eredmények értékelése

4.2.1. Konvergencia vizsgálata

A hálózat betanítása során célszerű megvizsgálni az előzőekben bemutatott veszteségfüggvények alakulását. Miként a 15. ábra is mutatja, a tanítás első lépéseitől kezdve a hibafüggvény értéke meredeken csökkenni kezdett. Ez a gyors csökkenés az első száz iteráció során még továbbra is tartott, ezt követően pedig lelassult. Jól látható az ábrán, hogy a tanítás sikeresnek mondható, hiszen a kezdeti meglehetősen nagy hibát követően gyorsan, már néhány száz iteráció után beállt egy jóval kisebb értékre. Ugyanez mondható el a validációs mintára számolt hibára is, a leállítás pillanatáig mindkét veszteségfüggvény tartós csökkenést mutat. Vagyis az általam tervezett neurális hálózat nem csak a tanítási mintát tudta megtanulni, hanem a számára még ismeretlen, új bemenő adatsorok esetében is jó eredményeket nyújt.



15. ábra. A mérhető átlagos abszolút hiba értékének alakulása a tanítás és validálás egyes iterációi során

4.2.2. Eredmények kvalitatív értékelése

A fenti tesztek pusztán azt mutatják, hogy a neurális háló sikeresen talált egy leképezést a bemenő és a kimenő függvények között. Viszont nem kerülhető el annak vizsgálata sem, hogy a hálózat alapján adott hőátadási függvények mennyire hasonlítanak az elvártakhoz. Az a szerencsés helyzet áll fenn, hogy ismertem az egyes lehűlésekhez tartozó hőátadási tényező függvényeket (hiszen ezek alapján generáltam a lehűléseket), így ez a vizsgálat egyszerűen megvalósítható. Össze tudtam hasonlítani a

lehűlési görbe generálásához használt referencia függvényt a neurális háló által (a lehűlési görbe alapján) adott függvénnyel. A 16. és az M34 - M36. ábrák bemutatnak néhány kiragadott példát a neurális hálózat által adott becslések közül. Az ábrán látható piros görbe mutatja a referencia (generált) hőátadási függvényt. A kék görbe mutatja az ehhez tartozó (neurális hálózat segítségével számolt) lehűlési görbét. Az alattuk levő zöld göbe pedig mutatja a referencia és a neurális hálózat (pusztán a lehűlési görbe alapján adott) válaszának abszolút különbségét. A zöld görbék elemzésével jól látható, hogy a módszer meglehetősen jól közelíti a valós eredményeket. Jól meghatározza az esetleges csúcspontokat és ezek esetleges értékét.

4.2.3. Eredmények kvantitatív értékelése

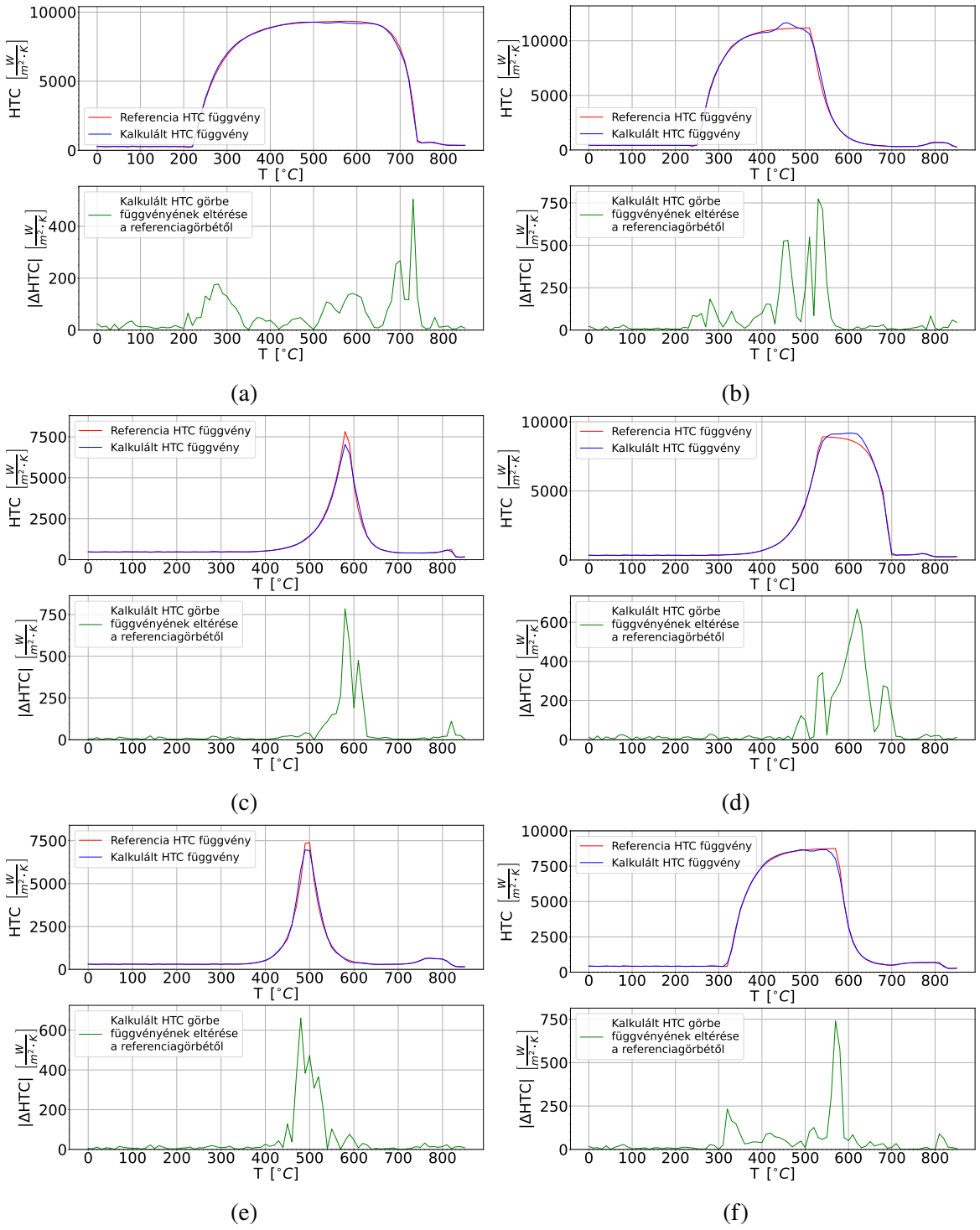
Az ábrákon jól látható, hogy a neurális hálózat megfelelő becsléseket tud adni. Az általam tervezett neurális hálózatot 759 iteráción keresztül tanítottam, amelyek közül a 708. iterációban érte el a legkisebb 0,00770 hiba értéket. A dolgozatban ismertetett neurális hálózat célja a leendő heurisztikus módszerek **indítási paramétereihez** megfelelő becslést adjon. Ennek okán nem elvárás a pontos becslés elérése. Mivel a leendő heurisztikus módszerek indításához szükséges véletlenszerű bemeneti adatok kiváltása a célja a neurális hálózatnak, célszerűnek látszik ezen véletlenszerű adatok alapján megítélni az eredmény megfelelőségét. Ebből a célból **egyesével** megvizsgáltam a validálási mintában található adatsorokat az alábbi lépéseket követve:

1. Beolvastam az eredeti hőátadási tényezőt, és az ehhez generált lehűlési görbét.
2. A betanított neurális hálózat segítségével lekérdeztem a hálózat becslését.
3. Generáltam egy teljesen véletlen hőátadási tényező függvényt.
4. Kiszámítottam a hálózat által becsült és a véletlen generálás által kapott függvények különbségét.
5. Ezek alapján elkészítettem a végső összehasonlító értékelést.

A vizsgálat eredményét a 2. táblázat tartalmazza. Az eredményekből jól látható, hogy az általam kidolgozott módszer két nagyságrenddel kisebb hibát eredményez, mint egy véletlen indítás. Ennek következtében várhatóan ezzel javulni fog az erre épülő heurisztikus keresések hatékonysága.

	Neurális hálózat által generált	Véletlenszerűen generált
Vizsgált minta száma	1000	1000
Átlagos abszolút hiba	$1{,}37 \cdot 10^4$	$1{,}25 \cdot 10^6$
Átlagos abszolút hiba szórásnégyzete	$3{,}51 \cdot 10^7$	$7{,}32 \cdot 10^9$

2. táblázat. Vizsgálat eredménye



16. ábra. A neurális hálózat által generált függvények összehasonlítása a referencia hőátadási függvényhez

4.2.4. Eredmények összehasonlítása hasonló publikációkkal

Az elérhető adatbázisokban 63 témához kapcsolódó cikk található, amelyekről azok tartalmi vizsgálata után kiderült, nagy része hőcserélő tartályok vizsgálatával foglakozik. Ezen publikációkban ismertetett neurális hálózatok bemeneti oldalán 3 – 7 mérhető fizikai paraméter, kimeneti oldalán 1 – 3

érték becslése a cél, feladattól függően. A 63 publikációból 2 darab publikáció ([86, 87]) kivételével egyik sem foglalkozott mért vagy számolt lehűlési görbéből hőátadási függvény becslésére tervezett neurális hálózatokkal.

A [87] publikációban egy darab eredmény ábra ([87] publikáció 4. ábrája), és mért hibafüggvények eredményei olvashatóak két táblázatban ([87] publikáció 2. és 3. táblázata). Ezen ábra alapján, és a hibafüggvények értékeiből, a jelen dolgozatban elért eredmények láthatóan pontosabbak, mint a publikációban közöltek.

A másik [86] (2023 év végén megjelent) publikációban ismertetett eljárás pontosságát a [86] publikáció 4. ábrája alapján becsülhetjük meg, ami azt jelenti, hogy körülbelül $3000\frac{W}{m^2 \cdot K}$ maximális ponttal rendelkező HTC függvényt a publikációban ismertetett neurális hálózat nem több, mint $500\frac{W}{m^2 \cdot K}$ átlagos abszolút hibával képes becsülni, ami azt jelenti, hogy maximum 16%-os átlagos abszolút hibával dolgozik. Az általam fejlesztett neurális hálózat a HTC függvényt maximum 11%-os átlagos abszolút hibával képes becsülni. Az ábrákról az is leolvasható, hogy a becslés hibája nem egyenletes, a legpontatlanabb a HTC függvény csúcspont körüli pontok, attól jobb és bal felé, a csúcsponttól távolodva a hiba mértéke jelentősen csökken. A [86] publikáció szintén más algoritmusok bemeneti paramétereinek közelítő becsléséről szól. A publikációban közölt eredmények alapján (feltehetően algoritmustól függően) bizonyítottnak tekinthető, hogy a neurális hálózat ilyen pontosságú becslése elegendő más, a HTC függvény becslésére szolgáló iterációs, heurisztikus algoritmus induló állapotának beállítására ([86] publikáció 6. ábrája).

*4.3. **2. Tézis***

Gépi tanulási modellt dolgoztam ki, amely az 1D végtelen hosszú tengelyszimmetrikus (hengeres) test felületén kialakult hőmérsékleti görbe alapján a felületi hőátadási együttható függvény becslésére alkalmas. A modell számítási pontosságát hipotetikus hőátadási függvények rekonstruálásán keresztül igazoltam.

4.4. Tézishez kapcsolódó saját publikációk

1. Sandor Szenasi, Zoltan Fried és Imre Felde: „Training of Artificial Neural Network to Solve the Inverse Heat Conduction Problem”. *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE, 2020. jan., 293–298. old. DOI: 10.1109/SAMI48414.2020.9108733.

5. PSO és FWA algoritmusok kiterjesztése

Számos szakirodalomi forrás – mint például [88, 89] – szerint a hengeres munkadarab HTC függvényének predikciójához jó választás lehet a PSO algoritmus használata. Több publikációban ([48, 90–92]) foglalkozom a kérdéssel, amelyek azt vetítik előre, hogy valós munkadarabon mért lehűlési görbék segítségével előállítható a HTC függvény.

5.1. Az általános részecske-raj optimalizációs algoritmus

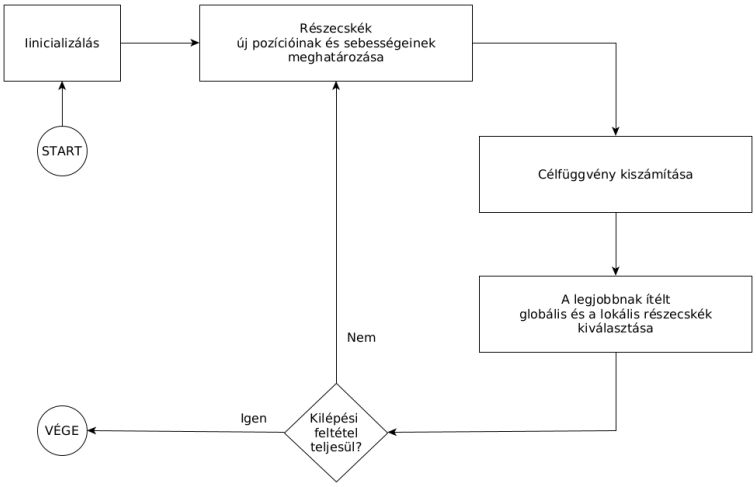
A részecske-raj optimalizációs algoritmust (PSO) James Kennedy és Russell C. Eberhart fejlesztették ki 1995-ben [50]. Az algoritmus alapját a madarak csoportjainak a mozgása adja. Az elmúlt majdnem 30 évben rengeteg variáns kidolgozására került sor, amelyek az algoritmus széles körű alkalmazását teszik lehetővé. Radha és társai [93] 2011-ben átfogó tanulmányt készítettek a PSO algoritmusok változatairól, és ezek lehetséges felhasználásairól. A PSO népszerűsége annak köszönhető, hogy fent tudja tartani az egyensúlyt a célfüggvény konvergenciája, és a bejárt út diverzitása között. Ez az eljárás a konvekciós és a sugárzási peremfeltételek rekonstrukciójához is alkalmas. Az eljárás hatékonyságát és eredményességét Vakil és Gadala [88] az inverz hővezetés analízis esetében vizsgálta. Ők az egyszerű, a„taszító", és a „teljesen taszító" PSO variációját használták az IHCP megoldásához 1, 2 és 3 dimenziókban. Az eredmények azt mutatják, hogy a PSO csökkentheti a klasszikus eljárások stabilitási problémáit az IHCP megoldások esetében.

Az általános PSO algoritmus két logikai építőelemből épül fel: a részecske és a raj. Egy részecske egy lehetséges megoldást reprezentál a keresési térben, míg a raj tartja össze a részecskéket, és irányítja azok mozgását. Minden részecske három információt hordoz, mint saját állapota. Ez az állapot áll a részecske pozíciójából, a pozíciójából számolt célfüggvény értékéből, és a részecske pozíciójának változását leíró sebességből. Ezen felül a részecskék rendelkeznek viszonylag rövid memóriával is a legjobb pozíciót illetően. A raj egyfajta globális tudatként tárolja az eddig legjobbnak ítélt pozíciót és célfüggvény értékeket. A legjobb pozíciót, a célfüggvénytől függően, a raj részecskéinek a számolt célfüggvényei közül a legkisebb vagy legnagyobb értéke adja.

A algoritmus röviden a következőképpen működik: a megfelelő számú részecske véletlenszerű generálása után meghatározzuk azok „megfelelőségét" – vagyis a célfüggvény értékeket –, majd kiválasztjuk egy optimálisnak vélt legjobb részecskét a számolt „megfelelőségéi" értékek alapján, valamint rögzítjük azt, mint globális legjobb értéket. A teljes raj mozgatását minden egyes részecskére számolt sebességéből kapjuk, majd az új pozíciókban ismét kiszámoljuk a célfüggvény értékeket. A folyamatot a 2. algoritmus és a 17. ábra mutatja be tömören.

2. Algoritmus A PSO algoritmus lépései

részecske paraméterek kezdeti értékének meghatározása ▷ $\mathcal{X}, \tilde{w}, c_1, r_1, c_2, r_2$
minden részecske helyének véletlenszerű kiválasztása
repeat
  for minden egyes részecskére **do**
    sebesség számolása
    a részecske új pozícióba mozgatása a sebesség alapján
    célfüggvény értékének számítása
    globál és lokál best értékének valamint egyéb szükséges paraméterek számítása ▷ $\vec{P}_i, \vec{G}_i, \ldots$
  end for
until kilépési feltétel ellenőrzése



17. ábra. A PSO algoritmus folyamatábrája

Az optimum keresése során a részecskék pozícióját és sebességét módosító képletek a következőek:

$$\vec{v}_{i+1} = \mathcal{X} \cdot [\tilde{w} \cdot \vec{v}_i + c_1 \cdot r_1 \cdot (\vec{P}_i - \vec{p}_i) + c_2 \cdot r_2 \cdot (\vec{G}_i - \vec{p}_i)] \tag{16}$$

$$\vec{p}_{i+1} = \vec{p}_i + v_{i+1} \tag{17}$$

ahol $c_1 \in \mathbb{R}$ és $c_2 \in \mathbb{R}$ a gyorsítási tényezők, értéküket szimulációs vizsgálatok útján szokás meghatározni, általában a nagyságuk egy 2 körüli érték. A c_1-et kognitív együtthatónak is szokás nevezni, amely annak a súlyát határozza meg, hogy a számítás során mennyire kell a lokális értékek felé tolni a részecske mozgását. Ez alapján a (16) egyenletből a $c_1 \cdot r_1 \cdot (\vec{P}_i - \vec{p}_i)$ részt kognitív résznek is hívják. A c_2 a szociális együttható, amely annak a súlyát határozza meg, hogy mennyire kell a globális legjobb érték felé tolódnia a részecske mozgásának. Emiatt a (16) egyenletből a $c_2 \cdot r_2 \cdot (\vec{G}_i - \vec{p}_i)$ tagot szociális résznek hívják. A (16) egyenletben a $\tilde{w} \cdot \vec{v}_i$ tag pedig az inercia rész. A $\vec{P}$ a lokális legjobb pozíciót, a $\vec{G}$ a globális legjobb pozíciót jelenti. Az $r_1 \in \mathbb{R}$ és $r_2 \in \mathbb{R}$ értékek $[0 \ldots 1[$ nyílt intervallumon értelmezett egyenletes eloszlású véletlen számok, amelyekkel biztosítjuk az algoritmus sztochasztikusságát, a $\tilde{w} \in \mathbb{R}$ és a $\mathcal{X} \in \mathbb{R}$ is súly értékek, amelyekkel a konvergencia sebességét lehet

szabályozni. A $\tilde{w}$ az úgynevezett inercia súly Shi és Eberhart [94] javaslata által, amely értékének meghatározásához képletek sokasága áll a rendelkezésre [54], de általánosságban azt lehet elmondani, hogy $\tilde{w} \leq 1$. A $\mathcal{X}$ [95] értékét a (18) alapján képezzük Clerc [96] javaslatára:

$$\mathcal{X} = \frac{2}{|2 - \varpi - \sqrt{\varpi^2 - 4\varpi}|} \tag{18}$$

ahol $\varpi = c_1 + c_2$ és $\varpi > 4$. Így a $\mathcal{X}$ értékét megközelítőleg 0,7298-ra szokás felvenni.

Különböző PSO variánsok elsősorban abban különböznek egymástól, hogy az egyes fent nevezett paramétereket milyen módon, és milyen értéken határozzák meg, esetleg az egyes részeket más tartalommal helyettesítik. Minden egyes PSO variáns valamilyen, de különböző probléma megoldásánál ad pontosabb és gyorsabb eredményt. Számos PSO variáns még plusz mutációvektort is alkalmaz (mint például a (19) [97] vagy a (20) [93]) a részecske pozíciók változékonyságának növelése érdekében:

$$\vec{v}_{\text{pk}} = \begin{cases} 0{,}5x_{\max} \cdot r_1 & \text{if } r_2 < 0{,}5 \\ -0{,}5x_{\max} \cdot r_1 & \text{egyébként} \end{cases} \tag{19}$$

ahol a $\vec{v}_{\text{pk}}$ egy véletlenszerűen választott részecske sebessége, a $\vec{v}$ a mutálandó sebességvektor, $x_{\max}$ a keresési tér felső határa, vagy

$$\vec{p}'_{i+1} = \vec{p}_{i+1} \cdot g \tag{20}$$

ahol $g \in \mathbb{R}$ egy standard normális eloszlású véletlen szám.

A gyakorlatban 3 különböző PSO variáns terjedt el. Az egyik népszerű változat a PSO inercia (PSO-In), amit akkor kapunk, ha a (16) egyenletben az $\mathcal{X} = 1$ behelyettesítéssel élünk. Ha a (16) egyenletben a $\tilde{w} = 1$ behelyettesítéssel élünk, akkor a PSO constriction (PSO-Co) változatot kapjuk. A harmadik nagyon népszerű változat a Quantum PSO (QPSO) [98–100]. Ennek mozgási egyenlete nagyon eltér a (16) alaktól, ugyanis elsősorban nem-lineáris problémák megoldásakor választják. A mozgási képletét a (21) egyenlet írja le:

$$\vec{p}_{i+1} = \vec{p}_i \pm \alpha \cdot |\vec{p}_i - \vec{\pi}_i| \cdot \ln\left(\frac{1}{u}\right), u, \alpha \in \mathbb{R} \tag{21}$$

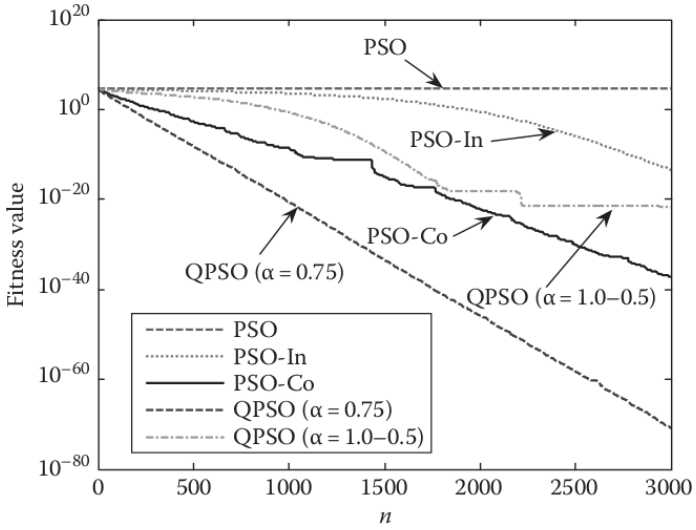
ahol az p_{i+1} az $i + 1$ iterációban a részecske pozíciója, u egy egyenletes eloszlású véletlen szám 0 és 1 között, az α egy úgynevezett összehúzó-elnyújtó (CE) paraméter, amely értékétől nagyon függ az algoritmus lefutása. Az értékét $[0, 1{,}781]$ között szokás felvenni [54]. A $\vec{\pi}_i$ a részecske helyi fókusza, amit a (22) egyenlet definiál:

$$\vec{\pi}_i = \varphi \cdot \vec{p}_i + (1 - \varphi) \cdot \vec{G}_i, \varphi \in \mathbb{R} \tag{22}$$

ahol $\varphi = (c_1 \cdot r_1)/(c_1 \cdot r_1 + c_2 \cdot r_2)$, valamint c_1, c_2, r_1, r_2, $\vec{p}_i$ és $\vec{G}_i$ megfelelnek a (16) egyenletben definiáltaknak.

A QPSO algoritmust [98–100] a kvantummechanika és a PSO részecskék pályájának elemzése ihlette [96, 98, 99], ahol a részecskék új pozícióját exponenciális eloszlás jellemzi. A keresési folyamatban adaptív stratégiát használ, az algoritmus a „delta potential well model”-en alapul, amely a Schrödinger egyenletből származik. Az algoritmust a [54] irodalom részletesen tárgyalja. Előnye, hogy nem szükséges sebességvektort számolni a részecskékhez, valamint az implementációja egyszerűbb, mivel kevesebb paramétert használ a hagyományos PSO-hoz képest.

A fenti variánsok (a PSO, a PSO-In, a PSO-Co és a QPSO) elnagyolt összehasonlításához vessünk egy pillantást a konvergencia sebességeket tartalmazó 18. ábrára.



18. ábra. A PSO, a PSO-In, a PSO-Co és a QPSO konvergencia sebességének összehasonlítása a sphere függvény megoldásakor [54]

5.2. Az általános Fireworks algoritmus

A Fireworks algoritmus [52] is egy raj-intelligencia algoritmus, amely lehetőséget ad nagyon nagy kiterjedésű és dimenziójú keresési térben végzett optimalizációra, miközben nagyban támaszkodik a véletlenszerűen megválasztott vizsgálandó pozíciók kijelölésére. Az algoritmus a tűzijáték szikráinak a robbanás központja körüli elhelyezésén alapszik, vagy más szavakkal a tűzijáték robbanási folyamatát követi, miközben mindössze csak két robbanási mintát implementál [52].

A Fireworks algoritmus (FWA) modellje „firework”-ből és több „spark”-ból épül fel. A firework jelenti a központi elemet, ami körül a robbanás történik, ahol spark-okat helyezünk el az $D \in \mathbb{N}$ dimenziós térben. Az i. spark-nak a pozícióját az D dimenziós keresési térben a $\vec{p}_i = (x_i^1, x_i^2, \ldots, x_i^D)$ vektor adja. Minden egyes spark egy potenciális megoldása lehet az IHCP problémának. Az algoritmus

épít arra, hogy a szimuláció során több populáció mozog egyszerre a térben, és ezek a populációk valamelyest hatnak egymásra. A terminológia szerint minden egyes populáció egy központi elemet és egynél több spark-ot tartalmaz.

A spark-okat a célfüggvény értékkel jellemezzük és hasonlítjuk össze egymással. A legjobb célfüggvény értékkel rendelkező spark mindig öröklődik a következő generációba, mint egy memóriatár hordozza a spark legjobb pozícióját és legjobb célfüggvény értékét a keresési térben. A populáció spark-jainak mozgását alapvetően két érték befolyásolja. Egyrészt az amplitúdó vektor, ami meghatározza, hogy az adott spark az aktuális pozícióból mekkorát léphet és milyen irányba, másrészt a lépéshez tartozó úgynevezett mutációs tényező. Minden egyes spark következő pozíciójához vezető lépésének nagyságát minden dimenzióban a következő egyenlet írja le:

$$\vec{d_i} = \vec{A_i} \cdot r_i \tag{23}$$

ahol az $i \in \{1, 2, 3, \ldots\}$ a spark indexe, $r_i \in \mathbb{R}$ egy egyenletes eloszlású véletlen szám -1 és 1 közötti nyitott intervallumon, az $\vec{A_i}$ az amplitúdó, és $\vec{d_i}$ az elmozdulás irányát és nagyságát jelző vektor. A sparkok mozgásának mutációjaként egy normál eloszlású véletlen számot is alkalmazhatunk a kiválasztott dimenziókban, amelyet a $g_i \in \mathbb{R}$ jelez 0 és 1 között, aminek a mean és variance értéke általában 1. Végül az új pozíció értékeit a

$$\vec{p}_{\text{new}_i} = (\vec{p_i} + \vec{d_i}) \cdot g_i \tag{24}$$

képlet adja. A spark-tól függően az d_i és g_i értékek közül az egyik elhagyható a mozgási egyenletekből. Az amplitúdó értéke függ az aktuális spark célfüggvény értékétől és a legjobb célfüggvény értéktől is:

$$\vec{A_i} = \vec{A}_{\max} \cdot \frac{S_i - S_{\min} + \varepsilon}{\sum_{i=1}^{N}(S_i - S_{\min}) + \varepsilon} \tag{25}$$

ahol az $\vec{A}_{\max}$ a maximális amplitúdó, ami a keresési térben előfordulhat, az $S_{\min} = \min(S_i)$ a legjobb célfüggvény érték az adott populációban. A sparkok száma a (26) egyenlet szerint alakul:

$$s_i = N_{\max} \cdot \frac{S_{\max} - S_i + \varepsilon}{\sum_{i=1}^{N_S}(S_{\max} - S_i) + \varepsilon} \tag{26}$$

ahol $N_{\max} \in \mathbb{N}$ a spark-ok maximális száma, $N_S \in \mathbb{N}$ a sparkok aktuális száma, az $S_{\max} = \max(S_i)$ a legrosszabb célfüggvény érték az adott populációban minden egyes sparkra (i), a ε egy kellőképpen kicsi szám a nullával való osztás elkerülésére.

Az amplitúdó minimális értéke is meghatározásra kerülhet adott esetben, annak érdekében,

hogy a spark következő pozíciója igazodjon ahhoz a tapasztalathoz, hogy a globális minimum az első iterációkban nagyon nagy valószínűséggel az aktuális pozíciótól távol található, idővel viszont a minimális célfüggvényű spark helye egyre közelebb lesz az aktuális spark helyéhez. Az $\vec{A}_{\text{min}}$ meghatározását a (27) adja:

$$\vec{A}_{\text{min}} = \vec{A}_{\text{init}} - \frac{\vec{A}_{\text{init}} - \vec{A}_{\text{final}}}{D_{\text{max}}} \cdot \sqrt{(2N_{\text{max}} - n)n} \tag{27}$$

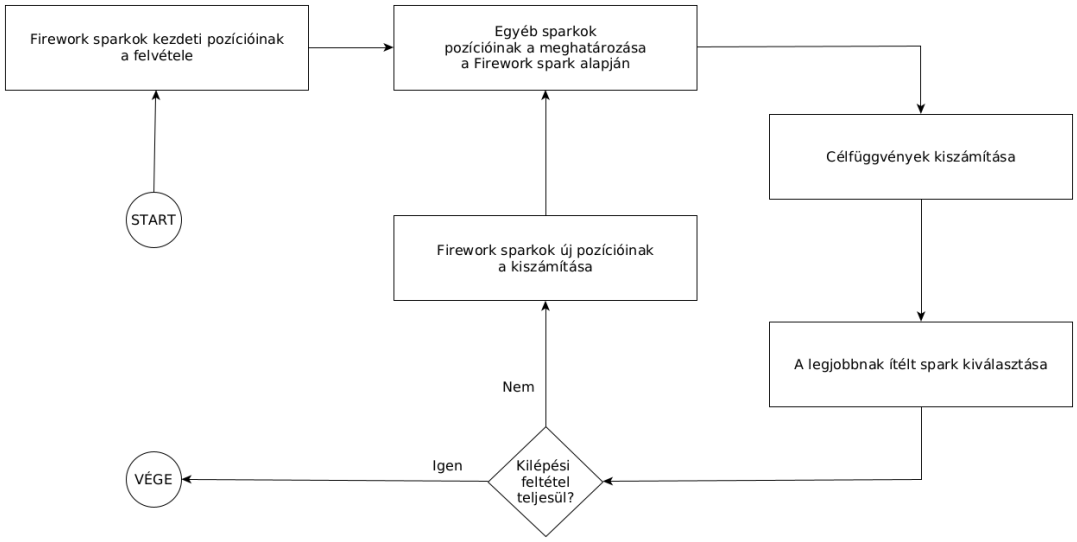
Minden iteráció után minden populációban minden spark célfüggvényértéke kiszámításra kerül, ezek közül a legkisebb érték (S_{best}) és az ahhoz tartozó spark pozíciója ($\vec{p}_{\text{best}}$) eltárolásra kerül.

Az FWA algoritmus egyszerűsített számítási lépéseit a legáltalánosabb formában a 3. algoritmus mutatja be:

3. Algoritmus Az FWA algoritmus lépései

```
firework-ök számának meghatározása
firework-ök helyének kiválasztása
repeat
    for minden egyes firework esetére do
        sparkok számának meghatározása
        minden spark helyének kiszámolása a spark mozgási stratégiája alapján
        célfüggvény számítása
        a legjobb és egyéb sparkok kiválasztása a választási stratégia alapján
    end for
until kilépési feltétel ellenőrzése
```

Erősen ajánlott párhuzamosított számítási eljárást használni a „célfüggvény számítása” lépésben, ami azért lehetséges, mert az egyes részecskék között ebben a lépésben nincs kommunikáció egy iteráción belül. Az FWA algoritmus diagramját a 19. ábra mutatja.



19. ábra. Az FWA algoritmus folyamatábrája

5.3. PSO és FWA algoritmusok felhasználása a hőátadási együttható függvény becslésére

A kereső algoritmusok sebessége a (8) egyenletben szereplő S célfüggvény értékének csökkentési sebességével is szabályozható. Ennek a sebesség növelésének ellentart az a feltétel, hogy az eredményül kapott HTC függvény például mennyire hasonlít egy valódi HTC függvényhez.

A (8) egyenletből az is látható (figyelembe véve a „reverse engineering" technika alkalmazását is), hogy a keresési térben bármilyen előfeltételezés nélkül semmilyen garancia nincs arra, hogy a (8) célfüggvény egyik *vélt* globális minimumának megtalálása esetén, a számolt $h(T^c(\vec{r},t))$ és $h(T^m(\vec{r},t))$ függvények azonos $\vec{r}$ és t pontjaira a

$$h(T^c(\vec{r},t)) - h(T^m(\vec{r},t)) = 0 \tag{28}$$

be fog következni. Ennek érdekében az 5.4. fejezetben egy olyan eljárást ismertetek, amely dinamikusan szűkíti a keresési tér tartományát egy olyan térrészre, ahol a számolt és mért HTC függvények kielégíteni igyekszik a (28) egyenletet. Az általam ismertetett új eljárást „puha" feltételként alkalmazom az algoritmus futása során, amivel az új eljárás „adaptív" módon járul hozzá az optimum megtalálásához.

Az új eljárással kiszámolt értéket, amely a (28) egyenlet teljesülés felé hajtja a PSO vagy FWA algoritmusokat, egy úgynevezett **„másodlagos" célfüggvényként** fogom alkalmazni a számítások során. Ez azt jelenti, hogy a $h(T^c(\vec{r},t))$ függvényt akkor tekintem kielégítőnek amikor az 5.6. fejezetben ismertetett feltételek teljesülnek.

Az 5.1. és az 5.2. fejezetekben ismertetett heurisztikus módszerek problémáinak kiküszöbölésével mindenképpen foglalkozni kellett. A keresést végző algoritmus hatékonyságának növelése érdekében szükséges, hogy az algoritmus futásának bármely pillanatában eldönthesse, hogy a számolt HTC görbe a keresési tér adott pontjában feltehetően lokális/globális minimuma van-e, és a számolt HTC görbe mennyire hasonlít egy valós HTC görbére. Amennyiben az algoritmus a fenti két tulajdonsággal rendelkezik, akkor – a vizsgálataim szerint – az algoritmus sokkal sikeresebben közelít egy *vélt* globális minimum felé a HTC görbe keresése közben, mint az 5.1. vagy az 5.2. fejezetben ismertetett algoritmusok bármelyike.

A populációban a legkisebb célfüggvény értékét képviselő részecske vagy spark a keresési tér egyes pontjaiban különböző állapotokba kerülhet. Ezen állapotok a következőek lehetnek:

- Lokális minimum: amikor a keresési tér egy pontjának közelében levő pontok mindegyikének a (8) célfüggvény értéke nagyobb, mint a vizsgált pont célfüggvény értéke.
- „Hamis" globális minimum: amikor a (8) célfüggvény értéke hibahatáron belül 0, és a (28) egyenletben szereplő feltétel nem telesül.

– „Valódinak tűnő” globális minimum: amikor a (8) célfüggvény értéke hibahatáron belül 0, miközben a (28) egyenletben szereplő feltétel teljesül.
– A fentiek közül egyik sem.

Természetesen az algoritmus a (28) egyenletben szereplő feltételt vizsgálni nem lesz képes, ezért e helyett meg kell elégednem a bevezetni kívánt másodlagos célfüggvény nagyságának a vizsgálatával. Ezen másodlagos célfüggvény nagyságának minimum értéke jelképezi majd a (28) egyenletben szereplő feltétel teljesülését.

A fenti 4 állapot bármelyike a futás során a rögzített telemetrikus adatokból ([101]), a (8) célfüggvény értékéből és az 5.4. fejezetben ismertetett értékből egyértelműen beazonosítható.

5.4. A másodlagos célfüggvény

A tapasztalataim azt mutatják, hogy az algoritmus gyorsabban konvergál egy várt optimum felé akkor, ha a mennyiségi adattal képes jellemezni a számított HTC függvény „megfelelőségét”, abban a tekintetben, hogy az egyes számolt HTC függvények mennyire realisztikusan közelítik meg a HTC függvényekre jellemző tulajdonságokat. Egy ilyen tulajdonság a (28) egyenletben szereplő feltétel is. A másodlagos célfüggvény értékének köszönhetően egymással összehasonlíthatóvá vállnak az egyes számított HTC függvények „megfelelősége”, de nem ad iránymutatást a 20. ábrán definiált függvénytől való éltérés mértékétől.

5.4.1. A másodlagos célfüggvény értéke

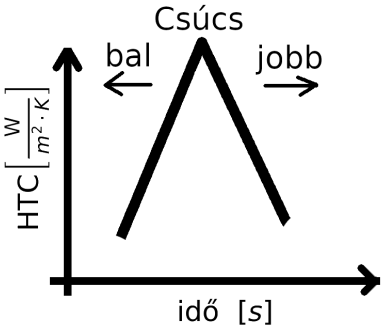
A számszerű jellemzéshez feltételezem, hogy a számolt HTC függvény egy darab maximum ponttal rendelkező függvény lesz, amelyre a maximumponttól balra levő pontjaira a függvény deriváltja pozitív vagy nulla, a maximumponttól jobbra levő pontjaira pedig a függvény deriváltja negatív vagy nulla, vagyis

$$\begin{aligned} h(T^c(\vec{r},t)) =& h(T^c(\vec{r},t))_b + h(T^c(\vec{r},t))_k + h(T^c(\vec{r},t))_j, \text{ amire} \\ & h(T^c(\vec{r},t))_b{}' \geq 0, \text{ ahol } b < m, \\ & h(T^c(\vec{r},t))_k{}' = 0, \text{ ahol } k = m \text{ és} \\ & h(T^c(\vec{r},t))_j{}' \leq 0, \text{ ahol } j > m \end{aligned} \tag{29}$$

ahol a b index a függvény m maximum pontjától *balra* elhelyezkedő pontokat jelöli, a j index a függvény m maximum pontjától *jobbra* elhelyezkedő pontokat jelöli, a k pedig a maximumpont helye.

Erre a speciális függvényre egy nagyon egyszerű példát a 20. ábra mutat. A másodlagos célfüggvény értékét úgy határozom meg, hogy a $h(T^c(\vec{r},t))$ függvény bármely pontjának bármilyen eltérése a a 20.

ábrán levő függvénytől a másodlagos célfüggvény nagyságát nullánál nagyobb mértékben befolyásolja. A másodlagos célfüggvény értékének a minimuma 0, és ez az érték jelenti azt az állapotot, amikor a számolt HTC függvény egy realisztikus HTC függvénynek tekinthető.



20. ábra. A számolt HTC függvény jellemzéséhez felhasznált függvény

$$h_{i+1} \geq h_i \tag{30}$$

ahol az $i \in \{1, 2, 3, \ldots, i_{\max}\}$ a HTC ($h \in \mathbb{R}$) függvény **bal** vagy **jobb** oldalán levő pontok indexe, az $i_{\max}$ a maximális HTC értékének az indexe a megfelelő irányból számolva.

Jelölje a másodlagos célfüggvény értékét az $S^\star$. Ennek az értékét a (30)–(31) egyenlet alapján úgy számolható, hogy ha a (30) feltétel nem teljesült a jobb- vagy baloldali ágra (egy adott pontban), akkor az adott oldalnak ($S^\star_{\textbf{jobb}}$ vagy $S^\star_{\textbf{bal}}$) értékét 1-gyel növelem. A végső $S^\star$ értékét a két oldal összegeként számolható. Ekkor

$$S^\star = S^\star_{\textbf{bal}} + S^\star_{\textbf{jobb}} \tag{31}$$

ahol

$$S^\star_{\textbf{bal}} = S^\star_{\textbf{bal}} + 1, \text{ ha } h_{i+1} < h_i \tag{32}$$

Természetesen itt az $i = 0$ pont a HTC függvény **bal** oldali részének a legalsó pontja, és ugyanígy az

$$S^\star_{\textbf{jobb}} = S^\star_{\textbf{jobb}} + 1, \text{ ha } h_{i+1} < h_i \tag{33}$$

esetén az $i = 0$ pont a HTC függvény **jobb** oldali részének a legalsó pontja. Az i értéke mindkét esetben a HTC függvény maximum pontjának eléréséig fut. Erre a számításra mutat egy példát a 4. algoritmus.

4. Algoritmus Példa a másodlagos célfüggvény értékének számítására

```
1  function CHECK_CONDITION(left_index, right_index)
2      return ( value of left_index > value of right_index )? 1 : 0
3  end function

4  function CALCULATE_SECONDARY_OBJECTIVE_FUNCTION(points of HTC function)
5      left_index ∈ ℕ ← 0    ▷ bal oldal legalsó pontjának indexe
6      right_index ∈ ℕ    ▷ jobb oldal legalsó pontjának indexe
7      peak_index ∈ ℕ    ▷ csúcspont indexe
8      result ∈ ℕ ← 0
9      for index ← left_index, . . . , peak_index - 1, increasing do    ▷ bal oldal számítása
10         result ← result + CHECK_CONDITION(index, index + 1)
11     end for
12     for index ← right_index, . . . , peak_index, decreasing do    ▷ jobb oldal számítása
13         result ← result + CHECK_CONDITION(index - 1, index)
14     end for
15     return result    ▷ másodlagos célfüggvény értéke
16 end function
```

5.4.2. A másodlagos célfüggvény csökkentése

Annak érdekében, hogy az algoritmus az $S^\star$ értékét a nulla felé hajtsa, az egyes részecskék/sparkok mozgását egy megfelelően választott – korlátozó – eljárással kellett kiegészítenem oly módon – függetlenül az alkalmazott optimalizálást végző algoritmustól (PSO vagy FWA) –, hogy a részecskék/sparkok pozíciója minden dimenzióban csak olyan irányban és nagyságban mozoghasson, hogy az $S^\star$ értékét – általában – csökkenni kényszerüljön. Továbbá a még hatékonyabb működés érdekében szükséges volt az algoritmust bővíteni azzal, hogy engedtem ezt a korlátozást (mutációként) megszegni azokban az esetekben, amikor ez a mozgás a HTC függvény szomszédos pontjainak (dimenzióinak) pozíciója miatt nem lehetséges. Ilyenkor először vizsgálom az ellenkező irányú mozgási lehetőséget, és ha az sem lehetséges, mert a részecske adott dimenziójának pozíciójában a szomszédos dimenziók (pontok) közé oly módon szorult, hogy nincs lehetősége semmilyen irányba mozogni, akkor szabadon megválaszthatta magának a lépése nagyságát a keresési tér határain belül (természetesen csak az adott dimenzión belül). Ez a korlátozó eljárás a következőképpen fogalmazható meg:

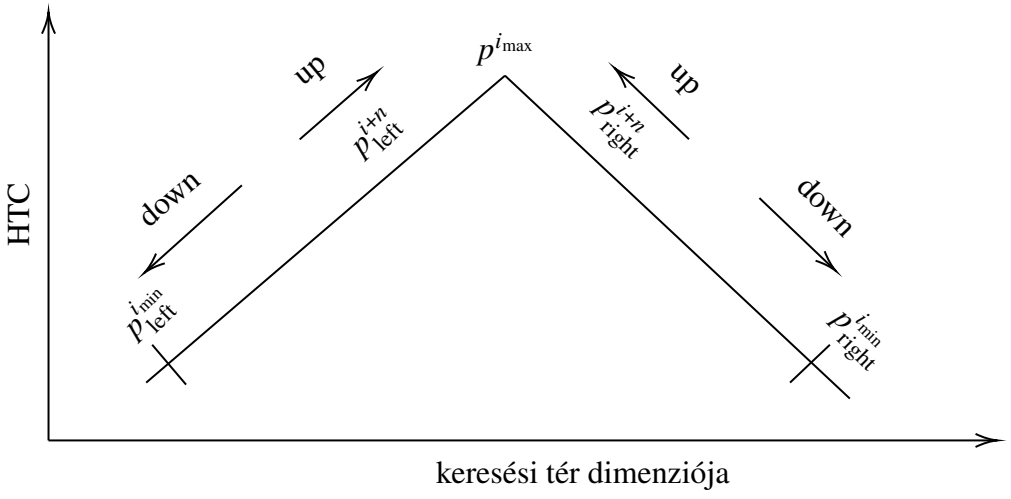
Jelölje a p^i a részecske i. dimenziójában az aktuális pozícióját, $p^{i_{\max}}$ a 20. ábrán a csúcspont pozícióját, p^0_{left} a 20. ábrán a **bal** oldali ág legalsó pontjának, p^0_{right} a **jobb** oldali ág legalsó pontjának a pozícióját, p^{i+n}_{left} a részecske i. dimenziójától jobbra levő $n \in \{0, 1, 2, \ldots\}$. dimenzió pozícióját, p^{i+n}_{right} a részecske i. dimenziójától balra levő n. dimenzió pozícióját, úgy, hogy ha $n = 0$, akkor az megegyezik p^i-vel, ha pedig $i = \max$, akkor az megegyezik a $p^{i_{\max}}$-el. A p^{i+n}_{right} és p^{i+n}_{left} pontokhoz hozzáillesztettem még az *up* vagy *down* jelzőket is, amivel jelezni lehet, hogy a p^i pontból a részecske lefelé vagy felfelé lépéséről beszélek a HTC függvény jobb vagy bal oldali ágán. Ezzel ki lehet jelölni egy lehetséges tartományt, hogy az adott részecske i. dimenzióját reprezentáló pont a következő lépésben hova várható. A felfelé mozgás esetében az *up* jelzőt, a lefelé mozgáshoz esetében a *down* jelzőt alkalmaztam.

$$A'^{i}_{\max_{\text{down}}} = \begin{cases} \textbf{\textit{bal}} \textit{ oldali ágra:} & p^i - p^{i+n}_{\text{left}} \\ \textbf{\textit{jobb}} \textit{ oldali ágra:} & p^i - p^{i+n}_{\text{right}} \\ \textbf{\textit{csúcspontra}}\textit{:} & \max\left(p^{i-1}_{\text{left}}, p^{i-1}_{\text{right}}\right) \end{cases} \tag{34}$$

és

$$A'^{i}_{\max_{\text{up}}} = \begin{cases} \textbf{\textit{bal}} \textit{ oldali ágra:} & p^{i+n}_{\text{left}} - p^{i}_{\text{left}} \\ \textbf{\textit{jobb}} \textit{ oldali ágra:} & p^{i+n}_{\text{right}} - p^{i}_{\text{right}} \\ \textbf{\textit{csúcspontra}}\textit{:} & p^{i_{\max}} \end{cases} \tag{35}$$

Vagyis a 20. ábra **bal** és **jobb** oldali ágára a részecske maximum $A^i_{\max_{\text{up}}}$-nyit léphet felfelé, és $A^i_{\max_{\text{down}}}$-t lefelé, ha nem engedem a keresési tér határainak az átlépését. Sajnos a (34) és a (35) egyenletekkel definiált eljárás csak akkor működik megfelelően, fz algoritmus által generált HTC függvény hasonlít a (30) egyenletben definiált függvényre. Általában nem ez a helyzet, hanem inkább hasonlít egy fűrészfog függvényhez. Emiatt módosítottam a (35) és a (34) egyenleteket a következőképpen:



21. ábra. Az $A^i_{\max_{\text{up}}}$ és $A^i_{\max_{\text{down}}}$ számításához szükséges jelölések jelentése

Mivel a HTC görbe pontjainak a kiértékelése a bal oldali ágon a csúcspontig balról jobbra, a jobb oldali ágon szintén a csúcspontig jobbról balra történik, az $A'^{i}_{\max_{\text{down}}}$ értékét a p^{i+n} helyett a p^{i-1} értékkel számolom, ugyanis az $i-1$. dimenzióban levő pontnál feljebb kell mozgatni a HTC görbe i. pontját. Az $A'^{i}_{\max_{\text{up}}}$ értékéhez is hozzá kell nyúlni, mivel előfordulhat, hogy a HTC görbe nem fordított V-re hasonlít, hanem V-re, és emiatt $A'^{i}_{\max_{\text{up}}}$ vagy $A'^{i}_{\max_{\text{down}}}$ negatív lesz. Ennek kiküszöbölésére a $\max(p^{i+n}_{\text{left/right}}, p^{i-1}_{\text{left/right}})$ maximumkiválasztást alkalmaztam. Abban az esetben, ha a számolt *up* pont kisebb lenne, mint a *down*, vagyis $A^i_{\max_{\text{up}}}$ vagy $A^i_{\max_{\text{down}}}$ negatív lenne, akkor az *up* pontot felemelem a

maximumra, hogy biztosan ne legyen alacsonyabban, mint a *down*, amivel plusz extra szabadságot adok a részecskének, hogy csökkenteni lehessen a másodlagos célfüggvényt, mint egy extra mutációs képesség. Ezzel a (35) és a (34) egyenletek a következőképpen módosulnak:

$$A^i_{\max_{\text{down}}} = \begin{cases} \textbf{\textit{bal}} \textit{ legalsó pont:} & p^{i_{\min}}_{\text{left}} \\ \textbf{\textit{jobb}} \textit{ legalsó pont:} & p^{i_{\min}}_{\text{right}} \\ \textbf{\textit{bal}} \textit{ oldali ágra:} & p^{i}_{\text{left}} - p^{i-1}_{\text{left}} \\ \textbf{\textit{jobb}} \textit{ oldali ágra:} & p^{i}_{\text{right}} - p^{i-1}_{\text{right}} \\ \textbf{\textit{csúcspontra}:} & \max\,(p^{i_{\max}-n}_{\text{left}}, p^{i_{\max}-n}_{\text{right}}) \end{cases} \tag{36}$$

és

$$A^i_{\max_{\text{up}}} = \begin{cases} \textbf{\textit{bal}} \textit{ legalsó pont:} & p^{i_{\min}}_{\text{left}} \\ \textbf{\textit{jobb}} \textit{ legalsó pont:} & p^{i_{\min}}_{\text{right}} \\ \textbf{\textit{bal}} \textit{ oldali ágra:} & \begin{cases} \max(p^{i-1+n}_{\text{left}}, p^{i+n}_{\text{left}}) - p^{i}_{\text{left}}, & ha\ \max(p^{i-1+n}_{left}, p^{i+n}_{left}) \geqslant p^{i-1}_{\text{left}} \\ p^{i_{\max}} - p^{i}_{\text{left}}, & ha\ \max(p^{i-1+n}_{\text{left}}, p^{i+n}_{\text{left}}) < p^{i-1}_{\text{left}} \end{cases} \\ \textbf{\textit{jobb}} \textit{ oldali ágra:} & \begin{cases} \max(p^{i-1+n}_{\text{right}}, p^{i+n}_{\text{right}}) - p^{i}_{\text{right}}, & ha\ \max(p^{i-1+n}_{\text{right}}, p^{i+n}_{\text{right}}) \geqslant p^{i-1}_{\text{right}} \\ p^{i_{\max}} - p^{i}_{\text{right}}, & ha\ \max(p^{i-1+n}_{\text{right}}, p^{i+n}_{\text{right}}) < p^{i-1}_{\text{right}} \end{cases} \\ \textbf{\textit{csúcspontra}:} & p^{i_{\max}}, \end{cases} \tag{37}$$

ahol $i_{\min} = 0$, $i_{\max} \geqslant i > 0$, $i_{\max} \geqslant n > 0$, és $i_{\max}$ a csúcspont index. Természetesen $i_{max} \geqslant i - 1 + n$ és $i_{max} \geqslant i + n$.

Az $A^i_{\max_{\text{up}}}$ és $A^i_{\max_{\text{down}}}$ számításánál az egyes jelölések jelentését a 21. ábra szemlélteti. Az algoritmus részecskéjének/spark-jának a tervezett i. dimenzióban a haladási iránya (up/down) egyértelműen definiálja, hogy melyik $A^i_{\max}$ értéket kell figyelembe venni az $A^i_{\max_{\text{up}}}$ vagy $A^i_{\max_{\text{down}}}$ közül. Mivel a PSO és FWA algoritmus is számol egy $A^{*i}_{\max}$ értéket, a végső $A^i_{\max_{\text{final}}}$ a

$$A^i_{\max_{\text{final}}} = \begin{cases} A^{*i}_{\max}, & ha\ A^{*i}_{\max} \leqslant A^i_{\max} \\ A^i_{\max} \cdot g, & ha\ A^{*i}_{\max} > A^i_{\max} \end{cases} \tag{38}$$

egyenlet szerint fog alakulni, ahol a $g \in \mathbb{R}$ egy 0 és 1 közötti nyílt intervallumban alkalmasan választott együttható. Az $A_{\max}$ ily módú meghatározására mutat példát a az 5. algoritmus. Az algoritmus belépési pontja a CALCULATE_NEW_OFFSET() függvény.

5. Algoritmus A másodlagos célfüggvény csökkentésének egy lehetséges, sematikus implementációja

```
function CALCULATE_MAX_AMPLITUDES(actual_position)  ▷ egy dimenzióban a HTC értékének csak felfelé vagy lefelé van lehetődége mozogni
    if is the position at the left bottom point? then  ▷ bal oldal legalsó pontja
        max_amplitude_up ← FIND_RIGHT_BIGGER_POINT(actual_position)
        max_amplitude_down ← GET_MIN_BOUNDARY()
    else if is the position on the left slope? then  ▷ bal oldal
        max_amplitude_up ← FIND_RIGHT_BIGGER_POINT(actual_position)
        max_amplitude_down ← FIND_LEFT_SMALLER_POINT(actual_position)
        if max_amplitude_up < max_amplitude_down then
            max_amplitude_up ← GET_MAX_BOUNDARY()
        end if
    else if is the position at the right bottom point? then  ▷ jobb oldal legalsó pontja
        max_amplitude_up ← FIND_LEFT_SMALLER_POINT(actual_position)
        max_amplitude_down ← GET_MIN_BOUNDARY()
    else if is the position on the right slope? then  ▷ jobb oldal
        max_amplitude_up ← FIND_LEFT_BIGGER_POINT(actual_position)
        max_amplitude_down ← FIND_RIGHT_SMALLER_POINT(actual_position)
        if max_amplitude_up < max_amplitude_down then
            max_amplitude_up ← GET_MAX_BOUNDARY()
        end if
    else  ▷ csúcspont
        max_amplitude_up ← GET_MAX_BOUNDARY()
        max_amplitude_down ← MAX(FIND_LEFT_SMALLER_POINT(actual_position), FIND_RIGHT_SMALLER_POINT(actual_position))
    end if
    return max_amplitude_up, max_amplitude_down
end function

function SELECT_MAX_AMPLITUDE(actual_position, direction)
    max_amplitude_up, max_amplitude_down ← CALCULATE_MAX_AMPLITUDES(actual_position)
    if is the direction up? then
        if max_amplitude_up > 0 then
            max_amplitude ← max_amplitude_up
        else  ▷ nem lehet felfelé lépni
            newDirection ← down
            max_amplitude ← max_amplitude_down
        end if
    else
        if max_amplitude_down > 0 then
            max_amplitude ← max_amplitude_down
        else  ▷ nem lehet lefelé lépni
            direction ← up
            max_amplitude ← max_amplitude_up
        end if
    end if
    if max_amplitude ⩽ 0 then  ▷ se le se fel nem lehet lépni
        if is the direction up? then
            max_amplitude ← GET_MAX_BOUNDARY() - actual_position
        else
            max_amplitude ← actual_position - GET_MIN_BOUNDARY()
        end if
    end if
    return max_amplitude, direction
end function

function CALCULATE_NEW_OFFSET(actual_position, offset)
    amplitude ← ABS(offset), direction ← SIGN(offset)  ▷ az offset tartalmazza a leendő lépés irányát és nagyságát egy adott dimenzió esetében
    mod_amplitude, mod_direction ← CALCULATE_AMPLITUDE(actual_position, direction)  ▷ a másodlagos célfüggvény csökkentése érdekében
    alkalmazandó maximális amplitúdó és irány meghatározása
    return mod_direction · mod_amplitude
end function
```

5.5. Mapping operátor/Mapping stratégia

A optimalizációs algoritmusok keresési terének határai „kemény” feltételeket (a 2.8.3. fejezet) támasztanak az algoritmussal szemben. Abban az esetben, amikor egy részecske következő lépésével kilépne az értelmezési tartományból, kell egy eljárás, ami meghatározza, hogy hova kerüljön ezen tartományon belül a lépés után, vagyis szükség van bizonyos leképezési módszerre. Ez algoritmusonként

különböző, nem létezik egy minden helyzetre tökéletes stratégia, egy olyan képlet, ami mindenfajta problémához ideális megoldást nyújtana. Egy széleskörűen használt stratégia például a rögzítéssel történő leképezés (Attract-Repulse mutation - [53]), ahol

$$p_i = p_i^{\min} + |p_i - p_i^{\min}| \ \% \ (p_i^{\max} - p_i^{\min}) \tag{39}$$

vagy egyszerűbben csak

$$p_i = p_i^{\min} + |p_i| \ \% \ (p_i^{\max} - p_i^{\min}) \tag{40}$$

ahol p_i egy részecske i. dimenzióbeli pozíciója, ami a keresési tér határain kívül esik, $p^{\min}$ és $p^{\max}$ a keresési tér határai, és a % jel a maradékos osztás művelete. Létezik olyan stratégia, ami azt mondja, hogy a keresési tér bármely véletlenszerű pontja tökéletesen megfelelő lesz ([102]):

$$p_i = p_i^{\min} + r \cdot (p_i^{\max} - p_i^{\min}) \tag{41}$$

Itt az $r \in \mathbb{R}$ egy egyenletes eloszlású véletlen szám. Ennek egy kicsit szofisztikáltabb változatára mutat példát a (42) képlet ([103])

$$p_i = \begin{cases} p_i^{\max} - r \cdot (p_i^{\max} - p_i^{\min}), & ha\ p_i > p^{\max} \\ p_i^{\min} + r \cdot (p_i^{\max} - p_i^{\min}), & ha\ p_i < p^{\min} \end{cases} \tag{42}$$

A (39) - (42) egyenletekben bemutatott mapping operátorokat széleskörűen használják az egyes algoritmusokban plusz mutációként is, amivel jogosan lehet az adott algoritmus diverzifikációját tervezetten növelni, ami az esetleges lokális minimumhelyek könnyű elkerülhetőségét segítik elő. Tapasztalatom szerint az IHCP algoritmus esetében ez a fajta működés hátráltatja a másodlagos célfüggvény konvergenciáját a nulla felé, mert a részecske pozícióját az a (39) - (42) egyenletek olyan irányba hajtják, hogy az eredményül kapott HTC függvény végül egy fűrészfog függvényhez fog hasonlítani.

Ezt elkerülendő, amikor a részecske a keresési tér egy adott dimenziójában átlépné a keresési tér határvonalát, akkor a határvonalig tartó lépés nagyságát változatlan irány mentén egy $(0, 1)$ értelmezési tartományú normál eloszlású véletlen számmal szorozva módosítom, úgy, hogy a normál eloszlású véletlen szám legvalószínűbb értéke 1 legyen. A normál eloszlású tényező miatt a következő pozíció inkább a határvonal közelébe fog kerülni, mint például az indulási pozíciójához közel, amivel ha el is romlik a másodlagos célfüggvény, de csak olyan mértékben, amit a haladási irány változatlansága miatt a (36) - (37) egyenletekkel könnyen korrigálható. Mindezt a (43) egyenlet szerint:

$$\vec{p}_{\text{new}} = \vec{p} \pm \vec{A}_{\max} \cdot g \tag{43}$$

ahol a $\vec{p}$ a részecske pozíciója, $\vec{A}_{\max}$ a lehetséges legnagyobb lépés nagysága és iránya a határvonalig, és a $g \in \mathbb{R}$ egy normál eloszlású véletlen szám, aminek a legvalószínűbb értéke esetén, a részecske pozíciója éppen a határra fog esni. Ezen elgondolás alapján a (38) egyenletben szereplő alkalmasan választott g együtthatónak is egy normál eloszlású véletlen számnak kell lennie.

5.6. Leállási feltétel

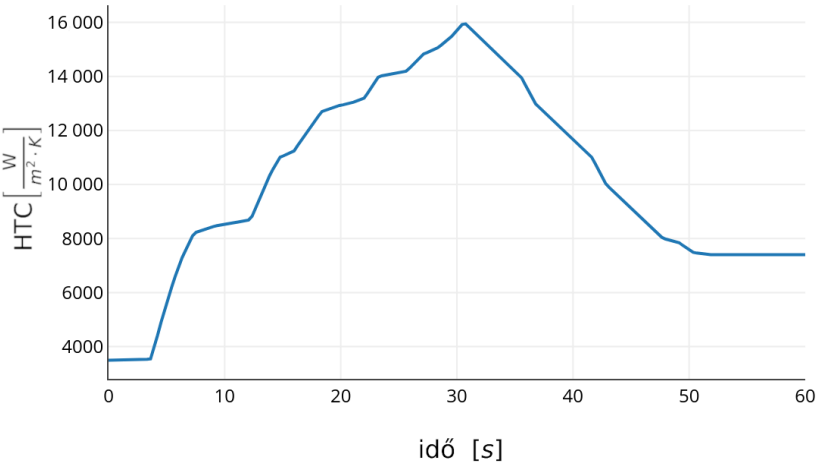
A populációk futása és egyben az algoritmus működésének a befejezése akkor következik be, amikor a célfüggvény értéke eléri, vagy kisebb lesz a minimálisnak beállított értéknél és a másodlagos célfüggvény értéke 0, vagy az algoritmus eléri az iterációk számának maximumát.

5.7. Az alkalmazott eljárás

A hőmérséklet eloszlás meghatározásához az 1D tengelyszimmetrikus hőátadási modellt alkalmaztam. Az inverz számításokat sorrendben a PSO, PSO-Co, PSO-In, QPSOT1 és QPSOT2 [54], valamint FWA [52], Adaptive Firworks Algorithm (AFWA) [104], Cooperative Firworks Algorithm (cFWA) [105], Enhanced Firworks Algorithm (EFWA) [102] és Enhanced Fireworks Algorithm with Differential Mutation (EFWADM) [106] algoritmussokkal végeztem. A keresett $h(t)$ függvény felbontását/pontjainak a számát 5-re, 50-re és 200-ra választottam annak érdekében, hogy különböző nehézségi terepen tudjam összehasonlítani az algoritmusok képességeit. A generált $h(t)$ függvény értékei 3500 és 16000 közöttre választottam. Azért, hogy összehasonlítható legyen az algoritmus különböző keresési terekben, a referencia $h(t)$ generált függvény mind az 5 esetben ugyanúgy néz ki (22. ábra), csak a meghatározandó pontjainak a száma változik. Mindegyik algoritmust 5 alkalommal futtattam le egymás után ugyanarra a keresési térre, **véletlenszerű indítási pozíciókból** indítva, és ebből választottam ki egy optimálisnak gondolt eredményt. A lehűlési görbék 600 pontból állnak. Végtelen hosszú 6,25mm átmérőjű hengeres munkadarab felületén mért hőmérsékletekkel dolgoztam, miközben a sugár irányába 96 egyenletesen elosztott pontban számoltam ki a hőmérséklet értékeket. A munkadarabot elméletileg 850°C-ra melegítettem és 20°C fokos hűtőközegben hűtöttem 60 másodpercig.

Annak érdekében, hogy a számolt HTC görbe abszolút hibáját meg lehessen határozni, az a legegyszerűbb módszer, ha teoretikus HTC görbével dolgozom, mely hasonló egy valódik HTC görbéhez. Emiatt a 22. ábrához látható görbét használtam mint referencia HTC görbe az eljárások ellenőrzésére.

Az egyszerűség kedvéért az elméleti HTC görbe is egy maximális csúcsponttal rendelkezik. Többfajta elméleti HTC görbére is futtattam vizsgálatokat. A kapott eredmények hasonló pontossággal és jellemzőkkel rendelkeztek.



22. ábra. A referencia HTC függvény

Az egyes PSO és FWA variánsok működésének helyességét a szakirodalmakban található teszt-függvényeken mint Sphere, Schwefel, Rosenbrock, Griewank, Ackley és Rastrigin algoritmusokkal ([107] 3.1.fejezet) ellenőriztem mint például: [108–112].

Az algoritmusok egyes paramétereit a 3., a 4. és az 5. táblázatok tartalmazzák.

Algoritmus	c_1	c_2	χ	$\tilde{w}$	α
PSO	2	2	1	0,5	-
PSO-Co	2,05	2,05	0,729	0,5	-
PSO-In	2	2	1	1,2 - 0,5	-
QPSOT1	-	-	-	-	1
QPSOT2	-	-	-	-	0,75

3. táblázat. PSO egyenletek paraméterei (a (16) és a (21) egyenletben szereplő tényezők, a [54, 97, 110–112] irodalmak alapján)

Keresési tér dimenziója	Populáció mérete	Iterációk maximális száma	Célfüggvény min. értéke (ε)
5	50	3000	0,1
50	100	3000	0,5
200	400	3000	1

4. táblázat. PSO algoritmusok futási paraméterei. Az algoritmusok nevei: PSO, PSO-Co, PSO-In, QPSOT1 és QPSOT2

Keresési tér dimenziója	Populáció mérete min.	max.	száma	Iteráció populáció	spark	Célfüggvény min. értéke (ε)
5	50	75	2	20	150	0,1
50	100	150	2	20	150	0,5
200	400	500	3	20	150	1

5. táblázat. FWA algoritmusok futási paraméterei. Az algoritmusok nevei: FWA, AFWA, cFWA, EFWA és EFWADM

Mindegyik PSO és FWA variánssal ugyanazt a mérési eljárást alkalmaztam. Mindegyik PSO és FWA variánssal futtattam szimulációt a szakirodalmakban vázolt algoritmus szerint az 5.4. és az 5.5. fejezetekben ismertetett kiegészítésekkel, és azok nélkül is az 5, 50 és 200 keresési tér dimenziókban. A következőkben minden szimuláció esetén megmutatom, hogy a szimuláció végén milyen és mekkora volt az eltérés a referencia és a számolt HTC és lehűlési görbe között, valamint a célfüggvény és a másodlagos célfüggvény iterációnkénti alakulását is. Vizsgálataim szerint a másodlagos célfüggvény alkalmazása a mapping operator nélkül és fordítva, különböző, de rosszabb futási eredményeket produkál, mint a kettő eljárás együtt, ezért minden esetben a két kiterjesztés együtt használom. Amennyiben a szimuláció közben az algoritmus elérte a minimális célfüggvény értéket és a másodlagos célfüggvény nagysága 0 volt, az aktuális iteráció végeztével leállt annak futása, vagy ellenkező esetben a megengedett legnagyobb iterációt is végigszámolva állt le.

5.8. Eredmények értékelése

A futási eredményeket a 23. - 28., valamint az M1 - M30. ábrák szemléltetik a következő közös magyarázattal: A bal felső ábrákon a HTC, a jobb felső ábrákon a célfüggvény és a másodlagos célfüggvény lefutása látható közösen az iteráció függvényében logaritmikus skálán. Alattuk a bal oldalon a HTC görbe abszolút hibájával, valamint jobb oldalon a lehűlési görbe abszolút hibája található.

*5.8.1. Eredmények a másodlagos célfüggvény és mapping operátor alkalmazása **nélkül***

Az 5 és 50 dimenziók esetében az ábrákon piros szaggatott vonallal azon számolt eredményeket ábrázoltam, amelyeket akkor kaptam, amikor a szimuláció során **nem** alkalmaztam az 5.4. és az 5.5. fejezetekben ismertetett eljárásokat az adott algoritmus esetében. Az eredmények azt mutatták, hogy miközben az $S^\star \ggg 0$, az $S < \varepsilon$. Ezekből az eredményekből arra a következtetésre jutottam, hogy a szakirodalomban található és általam vizsgált PSO és FWA variánsok alkalmatlanok a feladat megoldására nagyobb dimenziók esetében. Az eredmények nagyban függtek attól, hogy az algoritmusok

indítási pozíciója a keresési téren belül hova lett felvéve. Véletlenszerű indítási pozíciók esetében az általam vizsgált három dimenzió közül (5, 50 és 200) csak az 5 dimenziós keresési térben kaptam értékelhető eredményeket.

*5.8.2. Eredmények a másodlagos célfüggvény és mapping operátor alkalmazása **esetében***

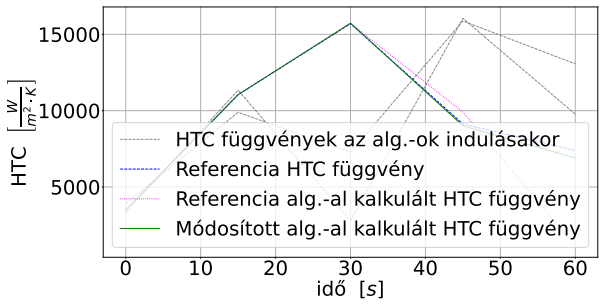
Az M2. és az M3. táblázatok a számszerű futási eredményeket tartalmazzák a PSO és FWA algoritmusok esetére. A táblázatok a következő szimulációs eredményeket mutatják be:

- Elvégzett kalkulációk száma.
- Célfüggvény minimum és maximum értéke, számtani átlaga, medián értéke, standard eltérése és standard hibája.
- Másodlagos célfüggvény nagysága (amikor meghatározásra került).
- Futási idő minimuma, maximuma, számtani átlaga és medián értéke.

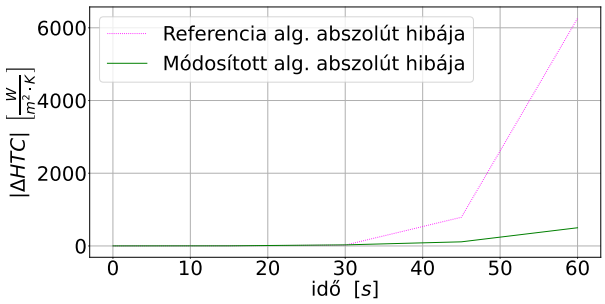
A szimulációk paramétereit a 4. és az 5. táblázatok tartalmazzák. A futási eredmények az M2. és az M3. táblázatok, a 23. - 28., valamint az M1 - M30. ábrák mutatják. Ezek alapján a következő megfigyeléseket tettem:

- A másodlagos célfüggvénnyel és mapping operátorral csak a HTC görbe jobb fele tért el – akár jelentősebben – a várt HTC függvénytől, a kapott függvény az elvártak szerint alakult. Ezen eltérés okainak kiderítéséhez további vizsgálatok szükségesek.
- A futási idők, a kalkulációk száma és a grafikus eredmények vizsgálata alapján a PSO és FWA algoritmus variánsok 5 dimenziós keresési térben könnyen és gyorsan eredményre jutottak, de 50 és 200 dimenziós keresési terek esetében jelentős eltérések mutatkoznak a referencia és a generált HTC görbék között.
- Az FWA algoritmusok esetében a szimulációk során kapott magas minimum célfüggvény értékek azt mutatják, hogy az adott algoritmusnak a maximálisan elvégezhető számítás sem volt elég a referencia lehűlési görbét megfelelő mértékben megközelíteni. Ha megnézzük ezen szimulációk futási idejét, látható, hogy bőven túl van az elvárható értékeken.
- A PSO algoritmusok elfogadható pontosságú eredményeket produkáltak elfogadható sebességgel.
- Az 5. táblázatban foglalt futási paraméterek választásakor az $S < \varepsilon$, valamint az $S^{\star} = 0$ feltétel teljesítése a generált HTC függvénynek a referencia függvényhez viszonyított pontosságát növelte, miközben a futási időt csökkentette.
- **A másodlagos célfüggvény és a mapping operátor teljesítették az elvártakat**.

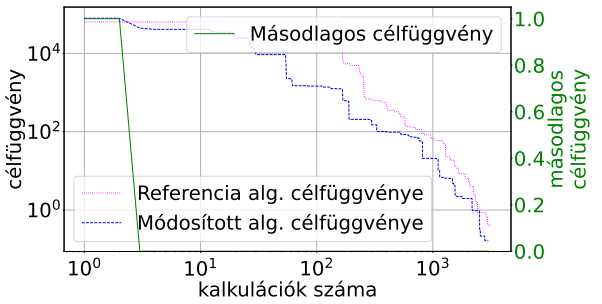
A módosított PSO és FWA algoritmusok által kapott eredmények a kidolgozott számítási eljárások hatékonyságát és alkalmazhatóságát támasztják alá.



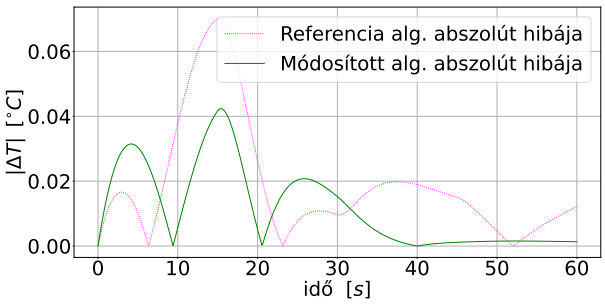
(a) Kalkulált HTC függvény



(b) Kalkulált HTC függvényének eltérése a referencia görbétől

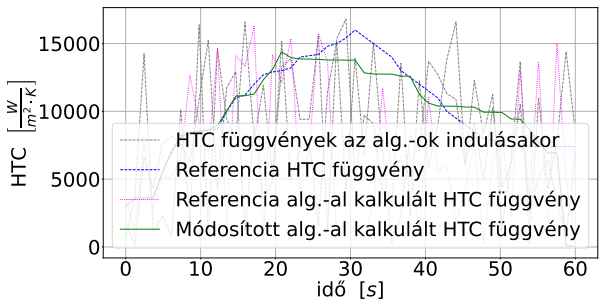


(c) Célfüggvény értékének változása

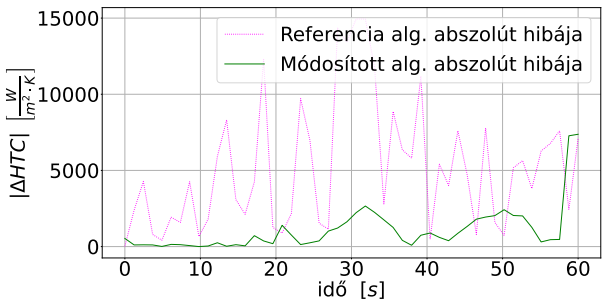


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

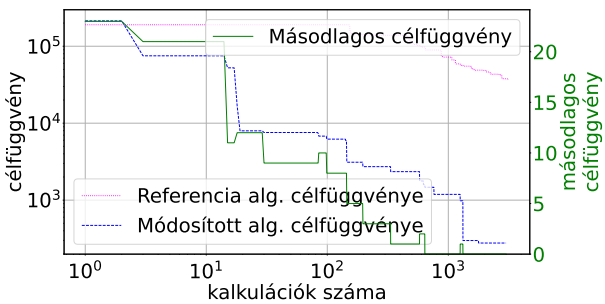
23. ábra. Az AFWA algoritmus futtatásának eredménye 5 dimenziós keresési térben



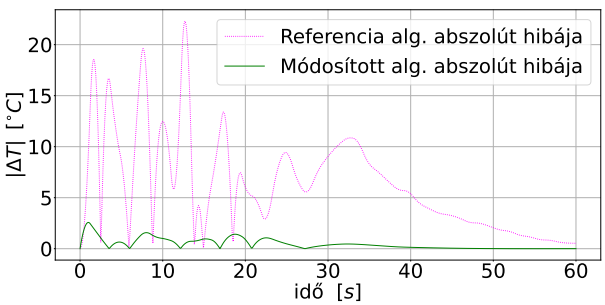
(a) Kalkulált HTC függvény



(b) Kalkulált HTC függvényének eltérése a referencia görbétől

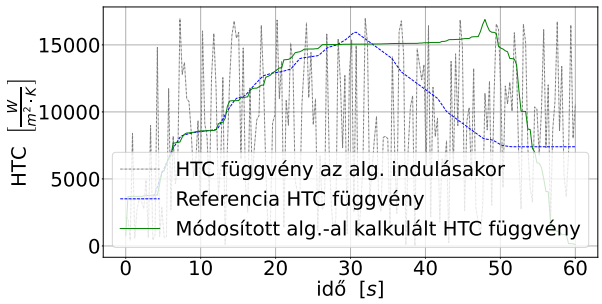


(c) Célfüggvény értékének változása

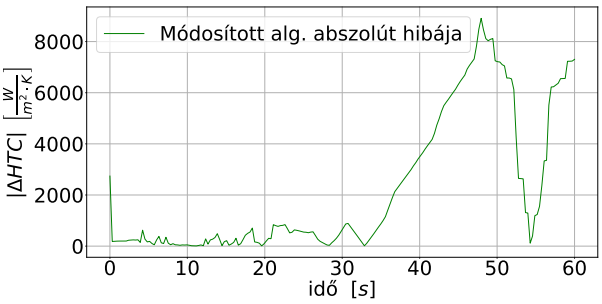


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

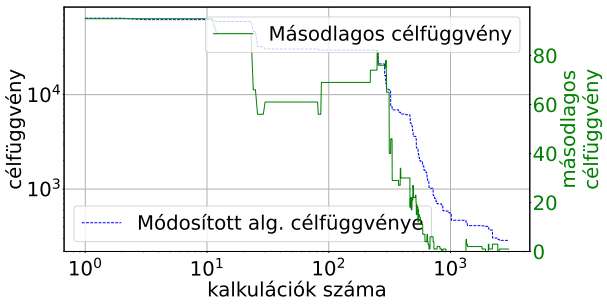
24. ábra. Az AFWA algoritmus futtatásának eredménye 50 dimenziós keresési térben



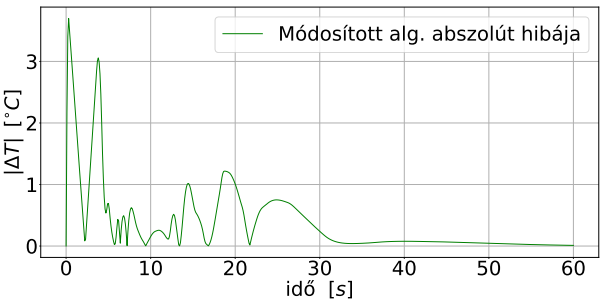
(a) Optimális spark HTC függvény



(b) Kalkulált HTC függvényének eltérése a referencia görbétől

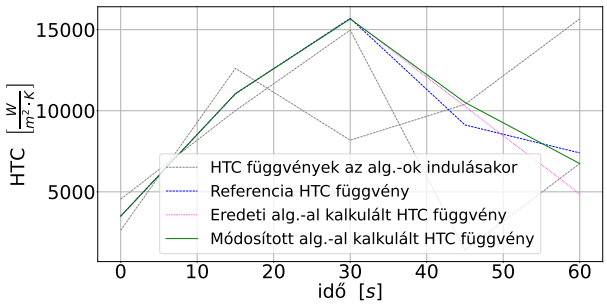


(c) Célfüggvény értékének változása

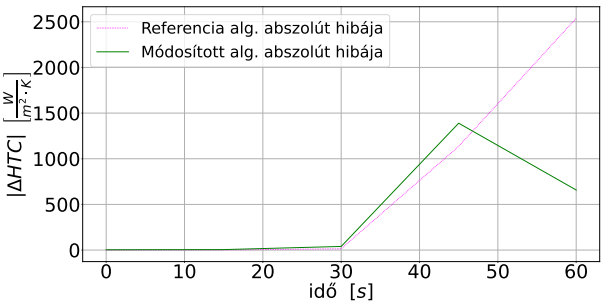


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

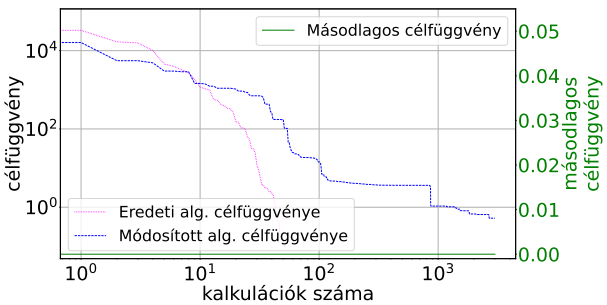
25. ábra. Az AFWA algoritmus futtatásának eredménye 200 dimenziós keresési térben



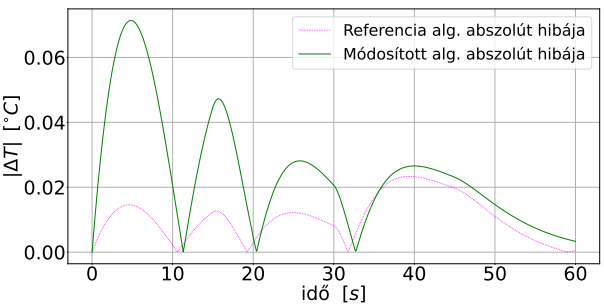
(a) Kalkulált HTC függvény



(b) Kalkulált HTC függvényének eltérése a referencia görbétől

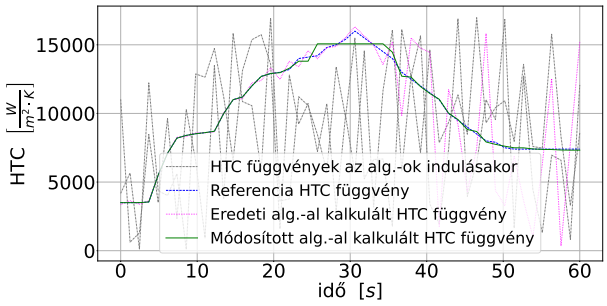


(c) Célfüggvény értékének változása

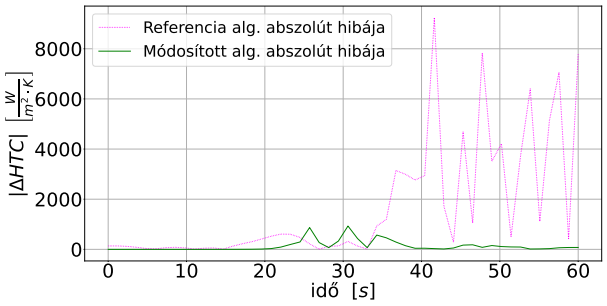


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

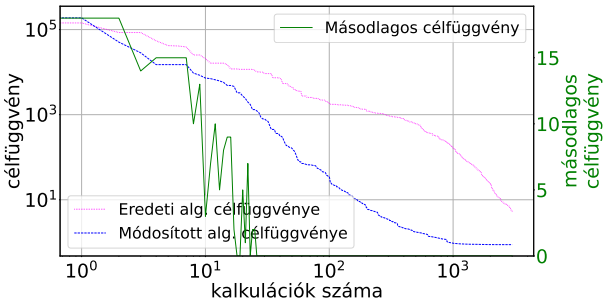
26. ábra. A QPSOT1 algoritmus futtatásának eredménye 5 dimenziós keresési térben



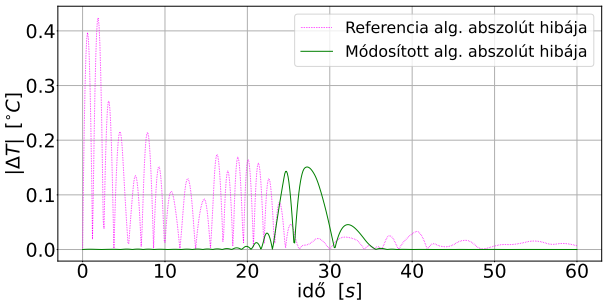
(a) Kalkulált HTC függvény



(b) Kalkulált HTC függvényének eltérése a referencia görbétől

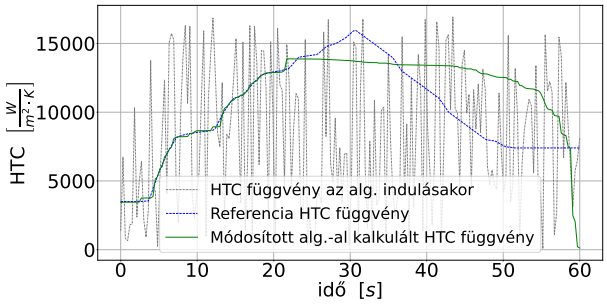


(c) Célfüggvény értékének változása

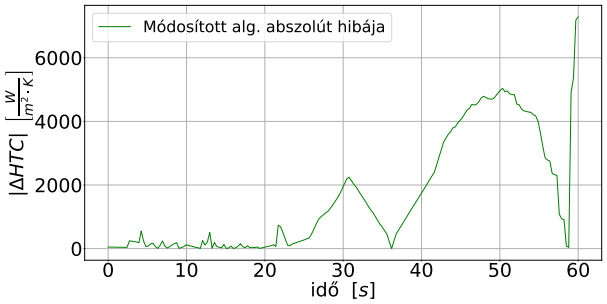


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

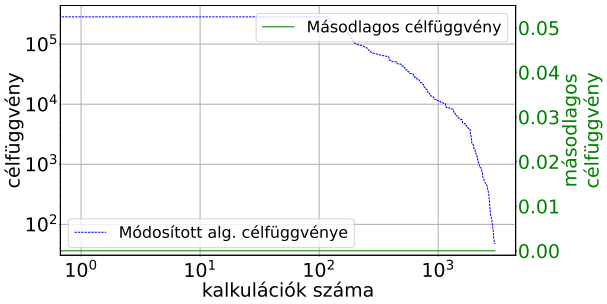
27. ábra. A QPSOT1 algoritmus futtatásának eredménye 50 dimenziós keresési térben



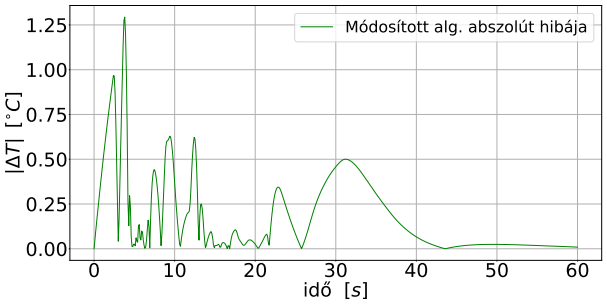
(a) Kalkulált HTC függvény



(b) Kalkulált HTC függvényének eltérése a referencia görbétől



(c) Célfüggvény értékének változása



(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

28. ábra. A QPSOT1 algoritmus futtatásának eredménye 200 dimenziós keresési térben

5.9. ***3. Tézis***

Új típusú megoldásokat fejlesztettem ki az 1D tengelyszimmetrikus test felületén kialakuló tranziens hőátadási együttható függvény becslésére felhasznált Részecske-raj Optimalizációs és Fireworks algoritmusokhoz. Az általam fejlesztett másodlagos célfüggvény és mapping operátor bevezetésével az inverz hőtani probléma megoldása a számítási pontosság növelésével és a futási idő csökkenésével járt a vizsgálatban bevont Részecske-raj Optimalizációs és Fireworks algoritmusok esetében a szakirodalomban közölt ugyanazon algoritmusokhoz képest 50 és 200 pontból álló hőátadási együttható függvények esetében, amelyeket szimulációs vizsgálatokkal igazoltam.

5.10. Tézishez kapcsolódó saját publikációk

1. Zoltán Fried, Sándor Szénási és Imre Felde: „Prediction of objective function value for heat transfer coefficient function reconstruction by FWA”. *2019 IEEE 13th International Symposium on Applied Computational Intelligence and Informatics (SACI)*. Timisoara, Romania: IEEE, 2019. máj., 305–308. old. ISBN: 978-1-7281-0685-4. DOI: 10.1109/SACI46893.2019.9111623.
2. Zoltán Fried és tsai.: „Parallelized Particle Swarm Optimization to Estimate the Heat Transfer Coefficients of Palm Oil, Canola Oil, Conventional, and Accelerated Petroleum Oil Quenchants”. *Materials Performance and Characterization* 8 (2018), 96–113. old. ISSN: 2379-1365. DOI: 10.1520/MPC20180049.
3. I. Felde, Z. Fried és S. Szénási: „Solution of 2-D Inverse Heat Conduction Problem with Graphic Accelerator”. *Materials Performance and Characterization* 6.5 (2017), 882–893. old. ISSN: 2379–1365. DOI: 10.1520/mpc20170008.
4. Zoltán Fried és tsai.: „Komplex hőátadási együttható rekonstrukciója bio-inspirált módszer alkalmazásával”. *XXVII. Hőkezelő és anyagtudomány a gépgyártásban országos konferencia és szakkiállítás külföldi részvevőkkel*. 2016, 53–58. old.

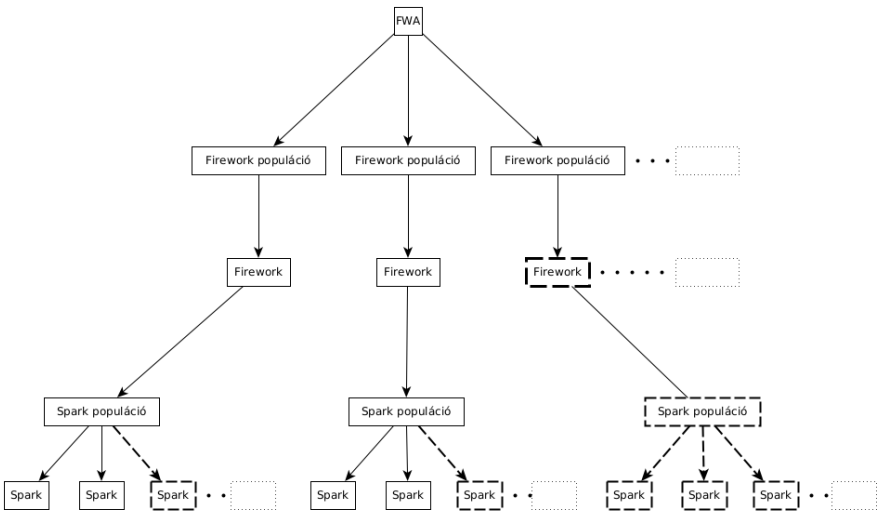
6. Új típusú FWA algoritmus

Az edzés közben kialakuló HTC függvény paramétereinek száma viszonylag nagy is lehet (> 100), így az optimalizálási problémát a paraméterek számának megfelelő dimenziójú térben szükséges megoldani. Mindehhez olyan optimalizáló módszer használata kívánatos, amely robusztus, elkerüli a lokális szélsőértékeket, ugyanakkor egyszerűen implementálható.

Az elmúlt évtizedben egyre gyakrabban alkalmazott optimalizálási módszer a FWA algoritmus, mely kielégíti az imént vázolt elvárásokat. A következőkben egy új típusú FWA algoritmust ismertetek, amelynek alapjaival az 5.2. fejezet foglalkozik.

6.1. Firework populáció

Az eredeti FWA modell (5.2. fejezet) a keresési terében – párhuzamosan – számos firework-öt futtatva igyekszik elkerülni a lokális minimumba ragadást (*a)* feltétel). Az új típusú FWA algoritmusban egy firework populáció egy fireworkból és a hozzá tartozó spark populációból, míg a spark populáció a firework körüli spark-ok halmazából áll (29. ábra). A szükséges firework populáció, vagyis firework spark-ok számáról elmondhatjuk, hogy függ valamilyen módon a keresési tér dimenzióinak számától. Felülről a futási idő növekedése korlátozza, alsó korlátnak értelemszerűen az 1 adódik.



29. ábra. Új típusú FWA algoritmus felépítése

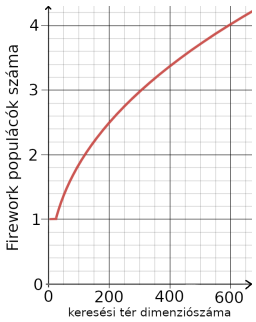
6.2. Firework populációk száma

Továbbiakban az egyszerűség végett mindegyik spark populáció egyenlő számú spark-kal dolgozik. Feltételezem továbbá, hogy minden egyes firework populációnak lehetősége van a teljes keresési teret átvizsgálására (*b)* feltétel). Az *a)* és *b)* peremfeltételek miatt arra lehet következtetni, hogy a firework

populációk száma a keresési tér dimenziószámától függ valamilyen függvény szerint. Kísérleti alapon született a (44) egyenlet:

$$N_p = \begin{cases} 1, & \text{ha } 25 > D \geq 3 \\ \frac{1}{4} e^{\lg(D)}, & \text{ha } D \geq 25 \end{cases} \tag{44}$$

ahol az $N_p \in \mathbb{N}$ a populációk száma az $\mathbb{R}^D$ ($D \in \mathbb{N}$ dimenziós) keresési térben. A függvény körülbelüli alakját a 30. ábra mutatja.



30. ábra. A Firework populációk számának javasolt függése a keresési tér dimenziószámától

6.3. Firework populációk és kölcsönhatásuk

A firework populációk kölcsönhatása azt jelenti, hogy az egyes párhuzamosan futó firework populációk információt cserélnek egymással arról, hogy a globális optimum keresése folyamán a keresési tér mely részét vizsgálják éppen, és ott milyen eredményre jutottak. A firework populációk ezen kommunikációt előre meghatározott iteráció elérésekor végzik el. Ekkor a firework populációk futása felfüggesztődik, megtörténik a firework populációk közötti információcsere, majd folytatódik azok futása. Az információcsere folyamán módosításra kerül minden spark populációban a best spark helye az összes spark populációban levő best spark pozíciójának súlyozott átlaga alapján, valamint minden egyes firework spark pozíciója is megváltozik a firework spark mozgási profilja alapján, ami a spark populációban található best spark közvetlen közelében levő pozíció jelent. Ezen súlyozott átlagot a (45). egyenlet, valamint a 6. algoritmus mutat be:

$$\mathfrak{p}_\mathrm{k} = \frac{1}{2}\left[p^2_{\mathrm{k=best}} + \frac{1}{D-1} \cdot \sum_{k=1,\text{ de } k\neq best}^{k=N_p} p^1_\mathrm{k}\right] \tag{45}$$

ahol $k \in \mathbb{N}$ a k. firework populációt jelöli ($1 \leqslant k \leqslant N_p$), p^1_k a firework populációban levő best spark pozíciója, $p^2_{\mathrm{k=best}}$ a firework populációban legkisebb másodlagos célfüggvény értékkel rendelkező sparkok pozíciója, $D \in \mathbb{N}$ és $D \geq 3$ a keresési tér dimenzióinak a száma.

6. Algoritmus Firework populációk kölcsönhatása

```
1  for each firework: selected_firework do
2      sum ← 0
3      for each firework: checked_firework do
4          if selected_firework = selected_firework then
5              sum ← sum + GET_SELECTED_FIREWORK_BEST2_POSITION() · (D − 1)   ▷ A kiválasztott firework populáció legkisebb másodlagos célfüggvény értékkel rendelkező spark pozíciója
6          else
7              sum ← sum + GET_CHECKED_FIREWORK_BEST_POSITION()   ▷ A kiválasztott firework populáció legkisebb célfüggvény értékkel rendelkező spark pozíciója
8          end if
9      end for
10     p_k ← sum/(2 · (D − 1))
11     SET_BEST_SPARK_POSITION(selected_firework, p_k)
12     SET_FIREWORK_SPARK_POSITION(selected_firework)
13 end for
```

6.4. Leállási/kilépési feltételek

A firework populációk futása, és egyben az algoritmus működése akkor fejeződik be, amikor a best spark másodlagos célfüggvény nagysága 0, és a célfüggvény értéke egy előre beállított érték alá csökken, – együttesen – bármelyik firework populációban, vagy az algoritmus elért egy előre meghatározott iterációs számot. Az algoritmusnak nem feladata az 0 célfüggvény érték elérése, ugyanis elég, ha a 0 közelébe jut, ahonnan például egy gradiens alapú megoldás könnyen pontosíthatja a számított HTC függvényt (7 fejezet).

A spark populáció iterációjának végét már nem ilyen egyszerű egy dinamikus értékkel meghatározni. A dolgozatban az iterációk száma egy fixen beállított 150 értékig futhat, mint legfelső határérték, vagy csak addig, amíg a best spark célfüggvény értékét csökkenteni képes, – ami a gyakorlatban azt jelenti, hogy a célfüggvény csökkenésének a sebessége [101] tartósan nem 0 –, illetve eléri a célfüggvény minimumaként meghatározott értéket. A végigszámolt iterációk száma a spark populáción belül a számítások során az első iterációkban sokkal alacsonyabb, mint az utolsó iterációkban, ugyanis a best spark célfüggvény értéke először gyorsan csökken, viszont az iterációk előrehaladtával egyre lassul a csökkenés mértéke, vagyis csökken a célfüggvény sebessége.

A spark populáció iterációjának vége után, ha firework populáció kilépési feltétele nem teljesül, akkor a firework populációk információt cserélnek egymással a 6.3. fejezetben ismertetett módon.

6.5. Amplitúdó nagysága és annak hatása

Az egyes spark-ok új pozícióinak számítása két folyamat eredménye. Az egyik folyamat azért felel, hogy megfelelő irányba hajtsa a populációt, a másik pedig a választott irány alapján a lehető legkevesebb lépéssel – vagyis a lehetséges legnagyobb lépésközzel – érjen el a keresett optimális pozícióba. A lépések iránya és nagysága együtt egy számolható vektort eredményez. Ez a két érték (irány és nagyság) szoros összefüggésben van egymással. Tapasztalataim alapján konvergenciát nem

mutató, vagy lassú algoritmust kapnék akkor, ha a vektor mindkét értékét irányítani szeretném. Ezért a továbbiakban a különböző típusú spark-ok esetében vagy csak a következő lépés irányát, vagy csak a nagyságát igyekszem a lehető legoptimálisabban befolyásolni. A következő lépés nagyságának az értékét a továbbiakban „amplitúdó”-nak hívom.

A spark-ok következő lépését az $\mathbb{R}^D$ (keresési) térben úgy lehetne a legegyszerűbben elképzelni, mint amikor valaki lép egyet a föld felszínén a 3 dimenziós (keresési) térben az egyik helyről a másikra. Ennek a lépésnek van egy minimális és egy maximális hossza, ami az esetek többségében előre, vagy valamennyire oldalra mutat. A minimális értére mutat egy példát a (27) képlet, túl nagyot sem tudunk lépni, lévén, hogy a lábunk hossza, a terepviszonyok, súlyunk és egyéb korlátozó tényezők befolyásolják, vagyis annak maximális értékét korlátozzák. Ugyanígy fogunk eljárni a spark-ok következő lépésének a meghatározásánál is. Annak a számszerűsített értékét, hogy mennyire jó, illetve jobb helyre lépett egy adott spark, a célfüggvény értékből és egyéb számolt, „mért” értékek segítségével érdemes kikövetkeztetni.

A következő lépés nagyságát (amplitúdóját) egy adott pozícióból rengeteg minden befolyásolhatja, de legfőképpen az, hogy melyik spark-ról van szó. Először szeretném bemutatni a spark-okat, hogy azok milyen lépési stratégiával – profillal – rendelkeznek:

- Firework spark: ez a spark a vezér, minden más spark arra megy, amerre ez a spark halad. A [101] cikkben bemutatott okok miatt ez a spark mindig szorosan a best spark mellé lép (6.6.5. fejezet).
- Explosion spark: a firework spark körül választ magának egy véletlenszerű helyet egy maximális távolságon belül (6.6.1. fejezet).
- Gaussian spark: ez a spark minden esetben választ magának egy másik spark-ot, és azt igyekszik követni. Ebben az esetben nem a lépés nagyságát, hanem annak irányát számolom (6.6.2. fejezet).
- Quantum spark: ez a spark a QPSO után kapta a nevét, ugyanis a mozgási profilját a quantum PSO egyenletei szabályozzák (6.6.3 fejezet).
- Best spark: ez a spark képviseli a populációt abban a tekintetben, hogy mindig oda lép, ahonnan a legközelebb lehet kerülni egy optimumhoz, vagyis a legkisebb célfüggvény értékkel rendelkező spark helyét fogja elfoglalni (6.6.4 fejezet).

6.6. A spark-ok

Az FWA algoritmushoz szorosan hozzátartozik a sparkok mozgási profiljaihoz köthető tulajdonságok ismerete is. Ebben a fejezetben az egyes spark típusok ezen tulajdonságait veszem sorra. A spark-ok

mozgását akadályozó kényszerítő tényezőket mint a másodlagos célfüggvényt és mapping operátort, az 5.4. és az 5.5. fejezetekben részletesen tárgyaltam, és azokat onnan egy az egyben itt is alkalmazom.

Az FWA algoritmus feltételezett konvergenciája nagyon erősen függ a spark következő lépéséhez szükséges amplitúdó becslésének a pontosságától is. Abban az esetben, ha ezt az amplitúdó értéket alábecsülöm, akkor az explosion spark-ok elenyésző hányadban tudnak csak hozzájárulni a célfüggvény csökkentéséhez, és ezáltal a konvergenciához szükséges tulajdonságaik érvényre juttatásához. Emiatt a gaussian, firework, quantum spark-ok adják majd a legkisebb célfüggvény értékeket.

A best spark-ot kivéve minden spark új pozíciójának a számításához három dolgot kell előre meghatározni:

- Bázis pozíciója: az a pozíció a keresési térben, ahonnan a következő lépés pozícióját számolom.
- Amplitúdó vektor maximuma ($\overline{A}_{\max}$): a következő pozíció számításához használt elmozdulás vektor maximális hossza.
- Véletlen számok: spark típusától és funkciójától függő, valamilyen eloszlással rendelkező véletlen számok, amelyek befolyásolják a lépés irányát és nagyságát.

Ezen paraméterek értékének számítását az adott típusú spark tárgyalásánál ismertetem.

6.6.1. Explosion spark

Az explosion spark következő lépésének nagyságát az $A_{\max}$ értéke felülről korlátozza, iránya pedig véletlenszerű.

Abban az esetben, ha az $A_{\max}$ értékét túl nagyra választom, akkor a következő lépéssel bejárható tér nagyságát növelem, ami adott számú spark esetében csökkenti az esélyét annak, hogy a következő lépéssel csökkenjen a célfüggvény értéke is. Abban az esetben ha az $A_{\max}$ értékét túl kicsire választom, akkor korlátozom a lehetséges lépés nagyságát, vagyis lassítom az algoritmust. Vagyis létezik egy ideálisnak vélt $A_{\max}$ érték, ami minden iterációban más és más. Ezek miatt az $A_{\max}$ értékének számítása kulcsfontosságú az adott lépésben az elérendő célfüggvényérték mihamarabbi megközelítésében.

A [101] irodalom alapján az optimálisnak vélt stratégia az, ha a legkisebb célfüggvény-érték a következő lépésben is tud csökkenni, még akkor is, ha csak nagyon kis mértékben. Ugyanis, ha a csökkenés nem következni be, akkor feltételezhető, hogy az algoritmus rossz irányba hajtotta a keresést, vagyis az adott ponton a keresés megrekedt. Emiatt vagy teljesen máshol kellene a cél felé haladni, vagy módosítani kell a keresési feltételeken, ami felesleges számításokhoz vezet. Természetesen a legkisebb célfüggvény-érték bármilyen kis csökkenése sem jelenti azt, hogy az algoritmus a globális minimum felé fog haladni.

Ezen feltételek miatt – az explosion spark esetében – adaptív módon érdemes számolni az $A_{\max}$ értékét. A következőkben ennek egy lehetséges megvalósítását ismertetem.

A jelölések egyszerűsítése végett ebben az alfejezetben mind az A (amplitúdó), v (lépés nagysága) és a p (spark pozíciója a keresési térben) paraméterek értékei bármilyen index-szel minden dimenzióra külön számolandó. Az $A_{\max}$ becslésekor felhasználhatóak az utoljára számított paraméterek. Egy adott explosion spark-hoz tartozó $A_{\max}$ számítását két részre bontottam abban a tekintetben, hogy az adott iterációban a best spark pozícióját egy explosion spark adja, vagy egy más típusú spark.

Abban az esetben, ha a best spark pozícióját egy explosion spark adja, akkor minden explosion spark ideálisnak vélt amplitúdójának maximális értékét egy egyszerű aránypárral közelítem: az előzőleg legkisebb célfüggvény értékkel rendelkező explosion spark $A_{\max}$ ($A_{\max}^{\text{lastBest}}$) értékét hasonlítom az adott explosion spark utolsó lépésének nagyságával (A_{real}) – amivel a jelenlegi pozícióba jutott –, valamint a best spark célfüggvény értékét (S_{best}) hasonlítom az adott explosion spark célfüggvény értékéhez (S). A két hányados szorzata egy olyan tényezőt képvisel, amely jellemző az „előző" lépés nagyságának megfelelőségére ($A_{\max}^{\text{coef}}$). A lokális minimumhely felismerhetőségéhez szükséges jellemzőket a [101] irodalomban tárgyalom.

A vázoltak alapján az $A_{\max}$ értékét dinamikusan befolyásoló tényező ($A_{\max}^{\text{coef}}$) a (46) egyenlettel fogalmazható meg:

$$A_{\max}^{\text{coef}} = \frac{A_{\text{real}}}{A_{\max}^{\text{lastBest}}} \cdot \frac{S_{\text{best}}}{S} \tag{46}$$

Az $A_{\max}^{\text{coef}}$ értéke tehát függ attól, hogy a célfüggvény értéke mennyivel csökkent, valamint, hogy mennyire sikerült a számolt $A_{\max}$ értékét megközelíteni úgy, hogy a célfüggvény értéke csökkenjen.

A következő lépésben alkalmazott $A_{\max}$ értékét a (47) egyenletet alapján számolom:

$$A_{\max}^{b} = m'_b \cdot A_{\max}^{\text{coef}} \cdot A_{\max}^{\text{lastBest}} \tag{47}$$

ahol az m'_b egy un. összehúzó-elnyújtó paraméter, értéke 0,8, ha az explosion spark lokális minimumhelyen tartózkodik, és 1,2, ha nem. Abból a megfontolásból változik az m'_b értéke, hogy, ha az explosion spark lokális minimumhelyen tartózkodik, akkor az explosion spark helye nem változott (best spark), tehát az előzőleg számolt $A_{\max}$ valószínűleg túl nagyra sikeredett, és ezért az $A_{\max}$ értékét csökkenteni kellene egy relatíve kis értékkel. Abban az esetben viszont, ha az explosion spark **nem** lokális minimumhelyen tartózkodik – vagyis sikerült csökkenteni a célfüggvény értékét – akkor érdemes az $A_{\max}$ értékét óvatosan növelni, hátha a következő lépésben egy még nagyobb lépéssel lehet az optimális pozíció felé lépni.

A (46) és a (47) egyenleteket alaposan megnézve az $A_{\max}^{\text{lastBest}}$ értékre valójában nincs is szükség,

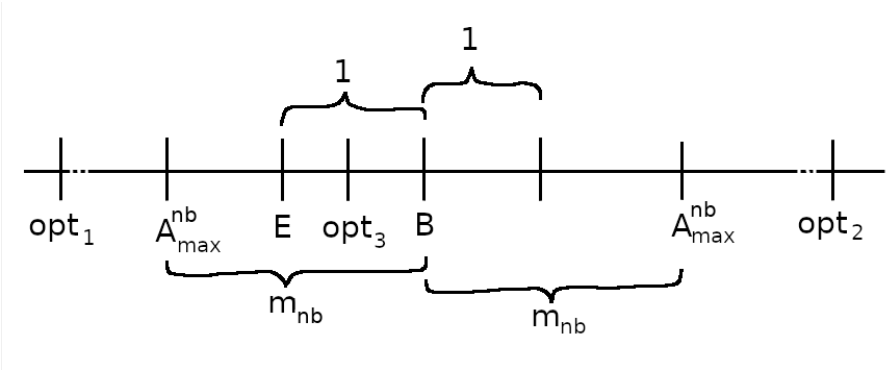
vagyis nem kell egy múltbeli értéket felhasználni az $A_{\max}$ kiszámításához. Így az $A_{\max}$ várható értéke a (48) egyenlet alapján becsülhető:

$$A^{b}_{\max} = m'_b \cdot A_{\text{real}} \cdot \frac{S_{\text{best}}}{S} \tag{48}$$

Abban az esetben, ha a best spark pozícióját **nem** egy explosion spark adja, akkor az $A_{\max}$ értékét az m_{nb} praktikus megválasztása mellett a best spark-tól számolt távolság alapján számolható, vagyis

$$A^{\text{nb}}_{\max} = m_{\text{nb}} \cdot \mid p_{\text{best}} - p \mid \tag{49}$$

ahol a p az explosion spark pozícióját, a p_{best} a best spark pozícióját jelenti. A m_{nb} értékét tapasztalati úton fixen 2-ben határoztam meg. A kutatás jelenlegi állása alapján a (49) egyenletben az m_{nb} értékéről annyit lehet biztosan kijelenteni, hogy nagyobbnak kell lennie, mint 1, ha el szeretném kerülni azt, hogy az explosion spark a best spark pozíciója felé konvergáljon minden esetben. Ez azért lenne előnytelen, mert korántsem biztos, hogy a best spark felé mutató vektor egyben az optimum felé mutató vektor is. A jelenséget a 31. sematikus ábra szemlélteti $\mathbb{R}^1$-ben:



31. ábra. m_{nb} nagyságának okai

ahol a B a best spark helye, a vizsgált E az explosion spark helye, az $A^{\text{nb}}_{\max}$ az E az explosion spark tapasztalati távolsága a best spark-tól, opt_1, opt_2 és opt_3 a lehetséges optimumok és $m_{nb} > 1$ távolság. Amennyiben a vizsgált explosion spark és a best spark 1 egységnyi távolságra vannak egymástól, akkor az $A_{\max}$-nak m_{nb} szeres távolságra kellene lennie a B best spark-tól.

Egyesítve az $A^{b}_{\max}$ és $A^{\text{nb}}_{\max}$ egyenleteket a

$$A_{\max} = m_{\text{nb}} \cdot \mid p_{best} - p \mid + m_b \cdot A_{\text{real}} \cdot \frac{S_{\text{best}}}{S} \tag{50}$$

kifejezést kapjuk, ahol az m_b értéke az m'_b értékei szerint alakul azzal a kiegészítéssel, hogy az m_b értéke 0 lesz, ha az explosion spark pozíciója **nem** esik egybe a best spark pozíciójával, vagyis az m_b

az (51) feltételekkel definiált:

$$m_b = \begin{cases} 0 & \textit{ha explosion spark pozíciója } \textbf{\textit{nem}} \textit{ esik egybe} \\ & \textit{a best spark pozíciójával} \\ 0{,}8 & \textit{ha a best spark-ot explosion spark adja és} \\ & \textit{a best spark lokális minimumhelyen tartózkodik} \\ 1{,}2 & \textit{ha a best spark-ot explosion spark adja és} \\ & \textit{a best spark } \textbf{\textit{nem}} \textit{ lokális minimumhelyen tartózkodik} \end{cases} \tag{51}$$

A fenti (50) egyenlet átalakítható a következők szerint:

– Az A_{real} kifejezhető a $\mid p^{i-1} - p^i \mid$ kifejezéssel.

– Az $A_{\max}$ a [101] cikkben definiált egyfajta sebesség.

– Az egyenletet egyenletes eloszlású véletlen számokkal kibővítve, és

– átrendezve a tagokat,

akkor az

$$A_{\max} = c_1 \cdot r_1 \cdot f_1 \cdot \mid p_{i-1} - p_i \mid + c_2 \cdot r_2 \cdot f_2 \cdot \mid p_i^b - p_i \mid \tag{52}$$

alakot kapom, ahol a c_1 értékei az (51) szerint alakulnak, $f_1 = \frac{S_{\text{best}}}{S}$, a $c_2 = 2$, $f_2 = 1$ és r_1 és r_2 egyenletes eloszlású véletlen számok a (23) egyenlet szerint, p_i az i. – vagyis aktuális iterációban a spark pozíciója, – p_i^b a best spark pozíciója i. aktuális iterációban.

Mivel az

$$A_{\max} = v_{i+1} - v_i \tag{53}$$

amit az (52) egyenletbe behelyettesítve

$$v_{i+1} = v_i + c_1 \cdot r_1 \cdot f_1 \cdot \mid p_{i-1} - p_i \mid + c_2 \cdot r_2 \cdot f_2 \cdot \mid p_i^b - p_i \mid \tag{54}$$

az (54) egyenlet hasonlít a (16) egyenlettel megfogalmazott PSO részecskék mozgását leíró egyenlethez. Ez arra utal, hogy sikerült kapcsolatot teremteni a PSO részecskék mozgási egyenlete és FWA explosion spark maximális lépésének a nagysága, más szóval annak mozgási egyenlete között.

Az explosion spark bázis pozíciója mindig a firework spark pozíciójával egyezik meg, vagyis az adott spark következő pozícióját a firework spark helyétől számolom.

6.6.2. Gaussian spark

Ezen spark típus új pozíciójának számítása egy normál eloszlású véletlen számon alapul (m = = 1, s^2 = 0,293), mint egy távolság a spark aktuális pozíciójától. Az elmozdulás iránya a best spark, vagy egy egyenletes eloszlású véletlen számmal választott másik spark poziciója a populációban. A kettő közül egy egyenletes eloszlású véletlen szám alapján dönt az algoritmus. A gaussian spark egyfajta mutációt valósít meg a best spark és egy választott spark között.

6.6.3. Quantum spark

Az FWA algoritmus és a keresési tér sajátosságai miatt gyakran előfordul, hogy a célfüggvény nem tud csökkenni, ugyanis az $A_{\max}$ nagysága miatt a kedvezőbb célfüggvény értéket túlságosan távol keressük, és az algoritmusnak sok iterációba telik ehhez alkalmazkodni. Ennek kiküszöbölésére – vagyis ezen alkalmazkodási folyamat felgyorsítására – implementáltam a QPSO-t (a (21) és a (22) egyenletekhez hasonlóan) mint egy spark fajta az FWA algoritmuson belül. Annak érdekében, hogy jól működhessen a QPSO az FWA-n belül, néhány módosítást végre kellett hajtanom a QPSO mozgási egyenleteiben minden egyes quantum spark-ra.

Egy quantum spark új pozícióját minden dimenzióban az (55) egyenlettel definiálom:

$$p_{\text{new}} = p_{\text{mix}} \pm A \tag{55}$$

ahol a p_{mix} a spark (22) által definiált pozíció, az A az amplitúdó.

Az (55) egyenletben az A értéke a (21) egyenlet szerint alakul, ahol az α értéke dinamikusan nő 20%-al ha a quantum spark célfüggvény értéke a best spark célfüggvény értékétől kevesebb mint 10%-ban tér el, egyébként az α értéke csökken 20%-al. De, ha quantum spark a best spark, akkor az α értéke nem változik. Az α kezdeti értéke 1. A (21) egyenletben a p a spark aktuális pozíciója, a X az összes quantum spark pozíciójának az átlaga, és az $u \in \mathbb{R}$ egy egyenletes eloszlású véletlen szám 0 és 1 között. A (22) egyenletben a G a best spark pozíciójának felel meg, mint global best, a P az összes quantum spark közül a legkisebb célfüggvénnyel rendelkező spark pozíciójának felel meg, mint local best, a $\varphi \in \mathbb{R}$ szintén egy egyenletes eloszlású véletlen szám 0 és 1 között.

6.6.4. Best spark

Ebből a spark típusból minden populációban egy létezik, és a helye mindig megegyezik az adott iterációban számolt legkisebb célfüggvény értéket birtokló spark pozíciójával. Más szóval, ez a spark hordozza a legkisebb célfüggvény értéket, ami továbbadódik a következő iterációba.

Több populáció esetén, a populáció iterációjának végén a best spark pozíciója módosul a minden populáció best spark-jának súlyozott átlagával. A számítás pontos menetét a 6.3. fejezetben fejtettem ki.

6.6.5. Firework spark

A [101] forrásban bemutatott elmélet alapján a firework spark helye a best spark helyéhez egy nagyon közeli pozíció. A firework spark $A_{\max}$ értéke a best spark-hoz való közelsége miatt minden dimenzióban megegyezik, és függ a best spark célfüggvény értékétől az (56) egyenlet szerint:

$$A_{\max} = \begin{cases} 1, & \textit{a kezdeti érték} \\ A_{\max}/10, & \textit{ha } 10A_{\max} > S_{\text{best}} \end{cases} \tag{56}$$

ahol az S_{best} a best spark célfüggvény értéke. Vagyis a firework spark maximális távolsága a best spark-tól egy nagyságrenddel kisebb, mint a best spark célfüggvény értéke. Az, hogy a firework spark pozíciója ilyen közel kerül a best spark-hoz, azzal jár, hogy a firework spark következő pozíciójában elég nagy esély van arra, hogy ő adja a következő best spark pozícióját is. Így valójában egy verseny alakul ki az Explosion, Gaussian, és Quantum spark-ok és a firework között. Amennyiben az Explosion, a Gaussian vagy a Quantum sparkok nem tudnak megfelelő célfüggvény értékcsökkenést elérni, nagy valószínűséggel a firework spark helye lesz a best spark következő pozíciója is. Ez azért előnyös, mert ezzel a stratégiával nagy valószínűséggel el lehet kerülni, hogy a best spark hosszabb ideig lokális minimumhelyen tartózkodjon.

6.7. Spark populációban résztvevő sparkok száma

A szimulációk során fontos érték a spark populációban résztvevő sparkok száma. Minden egyes spark típus esetében külön meg kell határozni azok számosságát. A 6.6. fejezetben ismertetett spark típusok közül a Firework és Best sparkok esetén algoritmikus okok miatt 1-re korlátoztam ezen sparkok számát. Pár futtatás után hamar kiderült, hogy az Explosion, Gaussian és Quantum sparkok számát érdemes a keresési tér dimenziójától függővé tenni. Tapasztalataim alapján a sikeres kereséshez szükséges sparkok száma – az általam vizsgált keresési tér tartományán belül ($5 - 1000$) – lineáris összefüggést mutatott a keresési tér dimenziószámával. Ezért az egyes spark típusok esetében is lineáris összefüggést feltételezek. Minimum értékek meghatározásához a következő összefüggéseket találtam:

Egy spark populációban a Gaussian spark-ok száma $N_g = \frac{2D}{3}$, Quantum spark-ok száma $N_q = \frac{2D-N_g}{3}$, az Explosion spark-ok számát pedig a $N_e = 2D - N_g - N_q$ képlet szerint javaslom felvenni, ahol a D a keresési tér dimenziójának a számát jelenti. Jobban megfigyelve a képleteket, az egyes spark típusok száma rendre ($N_e \in \mathbb{N}$, $N_g \in \mathbb{N}$ és $N_q \in \mathbb{N}$) a következőképpen alakul: $\frac{8}{9}D$, $\frac{6}{9}D$, $\frac{4}{9}D$.

Értelemszerűen ezen értékeket lehet magasabbra vagy alacsonyabbra is venni. A túl alacsonyra vett spark-számok ugyan kisebb futási időket jelentenek, mint az általam javasolt értékek alapján következne, viszont könnyen eltévedhet a program. Nagyobb értékek választása esetén nagyobb futási idők tapasztalhatóak, miközben a pontosság nem szükségszerűen nő. A „nagyobb futási idők" egyben nem jelentenek feltétlenül **arányosan** nagyobb futási időket is. Több spark nagyobb arányban tudja átvizsgálni a keresési teret, hamarabb eljut egy minimumba, ami várhatóan nagyobb eséllyel lehet a globális minimum környékén, mint ellenkező esetben, figyelembe véve a leállási feltételeket (6.4. fejezet) is.

6.8. Szimulációs paraméterek

A szimuláció indításához pár bemeneti paramétert meg kell adni. A fejezetben bemutatott szimulációkat az 5.7. fejezetben leírtaknak megfelelően készítettem el. A szimulációs eredmények összehasonlíthatósága érdekében figyeltem arra, hogy a sparkok száma egyezzen a FWA algoritmusok esetén használt sparkok, és a PSO algoritmusokban használt részecskék számával, aminek az értékeit az 5. táblázat tartalmazza. Az egyes sparkok számát a 6.7. fejezetben ismertetettek alapján számoltam (6. táblázat):

dim	Populáció mérete					száma	Maximum iteráció		Célfüggvény min. értéke
	best	firework	explosion	gaussian	quantum		fwa	spark	
5	1	1	23	16	10	2	20	150	0,1
50	1	1	44	33	22	2	20	150	0,5
200	1	1	178	133	88	3	20	150	1

6. táblázat. Szimulációs paraméterek

Az új típusú FWA megvalósítására a 7. algoritmus mutat egy példát.

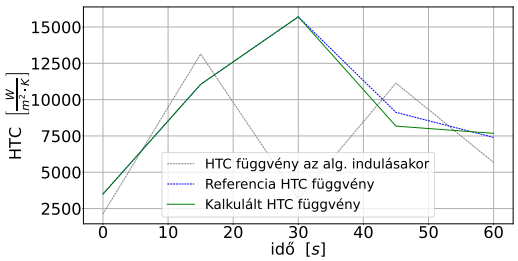
6.9. Szimulációs eredmények

Az általam ismertetett új típusú FWA algoritmus megfelelőségének megítéléséhez célravezetőnek tartottam az 5.1. és az 5.2. fejezetekben ismertetett általános PSO és FWA algoritmusokhoz való hasonlítást, mivel az 5 dimenziós keresési tér kivételével a szakirodalomban rögzített PSO és FWA algoritmusok futási eredménye értékelhetetlen. Ezen algoritmusok futási eredményeit az 5.8. fejezetben tárgyalom.

Amennyiben a szimuláció közben az algoritmus elérte a minimális célfüggvény értéket, az aktuális iteráció végeztével leállította az algoritmus futását, vagy ellenkező esetben a megadott iterációkat végigszámolva állt le. A számszerű eredményeket a 7. táblázat tartalmazza.

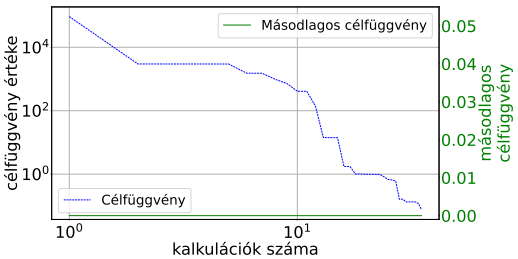
7. Algoritmus Új típusú FWA működése

```
firework populációk számának a meghatározása
for minden egyes firework populációban do
    firework spark pozíciójának véletlenszerű felvétele
    minden egyéb spark pozíciójának véletlenszerű felvétele a firework spark körül annak mozgási profilja alapján
end for

repeat
    repeat
        for minden egyes firework populációban do
            célfüggvény értékének meghatározása
            másodlagos célfüggvény értékének meghatározása
            best spark áthelyezése a legkisebb célfüggvény értékkel rendelkező spark pozíciójába
            minden egyéb spark következő pozíciójának meghatározása annak mozgási profilja alapján
            sparkok mozgatása az új pozícióikba
        end for
    until spark populáció kilépési feltételének ellenőrzése

    for minden egyes firework populációban do
        best spark új pozíciójának meghatározása a spark populációk közti információcsere segítségével
    end for
    for minden egyes firework populációban do
        best spark mozgatása az új pozícióba
        minden egyéb spark következő pozíciójának meghatározása annak mozgási profilja alapján
        sparkok mozgatása az új pozícióikba
    end for
until firework populációk kilépési feltételének ellenőrzése
```

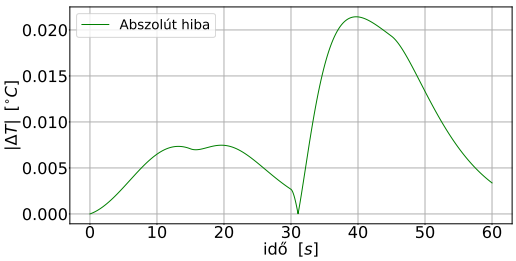


(a) Kalkulált HTC függvény

(b) Kalkulált HTC függvényének eltérése a referencia görbétől



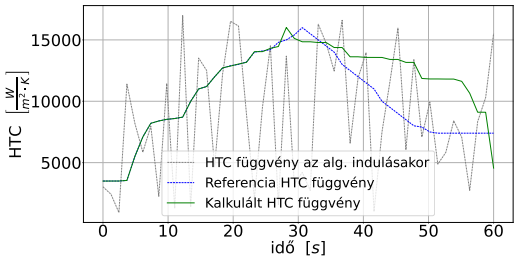
(c) Célfüggvény értékének változása



(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

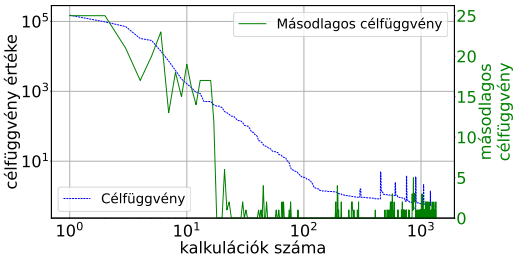
32. ábra. Új típusú FWA algoritmus futási eredménye 5 dimenziós keresési térben

Az általam fejlesztett FWA algoritmus szakirodalmakban található tesztfüggvényeken mérhető megfelelőségét a dolgozatban szándékosan nem vizsgáltam, ugyanis az algoritmus nem azon függvények megoldására született, hanem kifejezetten a HTC görbe meghatározása a célja.

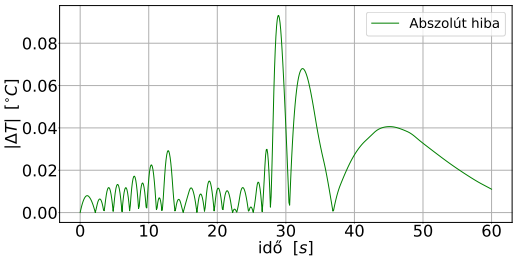


(a) Kalkulált HTC függvény

(b) Kalkulált HTC függvényének eltérése a referencia görbétől

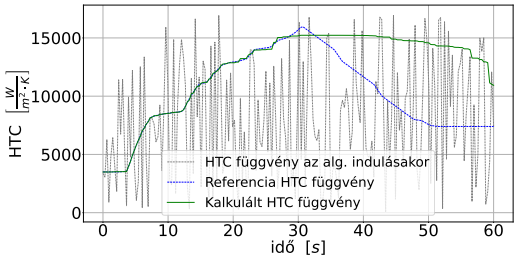


(c) Célfüggvény értékének változása



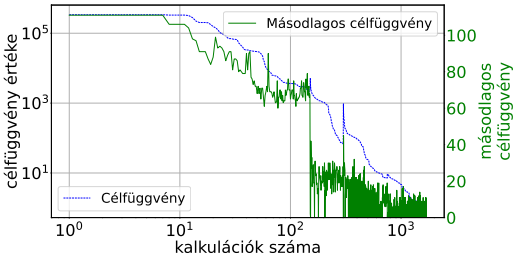
(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

33. ábra. Új típusú FWA algoritmus futási eredménye 50 dimenziós keresési térben

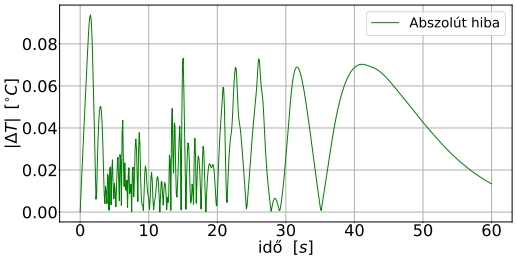


(a) Kalkulált HTC függvény

(b) Kalkulált HTC függvényének eltérése a referencia görbétől



(c) Célfüggvény értékének változása



(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

34. ábra. Új típusú FWA algoritmus futási eredménye 200 dimenziós keresési térben

A szimulációkhoz az a 22. ábrán látható elméleti referencia HTC görbét használom fel az 5.7. fejezetben foglat geometriájú munkadarab felületén számolt teoretikus *referencia* lehűlési görbe meghatározásához, és a szimuláció végén az eredmények értékeléséhez.

A szimulációs eredményeket a 32.-34. ábrák szemléltetik, ahol *a*) ábrán a zöld folyamatos vonal

jelöli a kalkulált HTC görbét, a referencia HTC görbét a kék szaggatott görbe mutatja. A referencia és kalkulált görbe közti abszolút különbséget *b*) ábra szemlélteti. Jól látható, hogy a csúcspontig a kettő különbsége elhanyagolható mértékű, viszont a csúcsponttól jobbra levő értékek jelentősen különböznek az elvárt értékektől. A *c*) ábrán a célfüggvény és másodlagos célfüggvény lefutása, a *d*) ábrán pedig a számolt és a referencia lehűlési görbe abszolút hibája látható.

6.10. Eredmények értékelése

Az eredményeket bemutató 7. táblázat, a 32 - 34. ábrák alapján futási időben elértem a kitűzött célt, a nagyobb pontosság érdekében további vizsgálatokra van szükség. Ez jelentheti akár egy gradiens alapú módszer alkalmazását is, mint amelyet például a 7. fejezetben ismertetek.

A 32 - 34. ábrákból látható, hogy a HTC függvény bal oldali ága a HTC függvény első 20 másodpercében 200 $\left\lfloor \frac{W}{m^2 \cdot K} \right\rfloor$-nál kisebb a referencia HTC-től való eltérése, de körülbelül 20 másodperctől kezdődően az eltérés mértéke egészen 7000 $\left\lfloor \frac{W}{m^2 \cdot K} \right\rfloor$-ig növekszik. Ez a tendencia megfelel a módosított FWA és PSO algoritmusoknál tapasztaltakhoz, vagyis ez a jelenség nem függ a keresést végző algoritmustól.

Az új típusú FWA algoritmus sebességben gyorsabb mint a módosított FWA algoritmus bármelyike, minden vizsgált dimenzió esetében, és nagyobb (200) dimenziós keresési tér vizsgálatakor a módosított PSO algoritmusoknál is hatékonyabb. A másodlagos célfüggvény az új típusú FWA algoritmus esetében is megfelelően működött.

Az 5., 6. valamint a melléklet I., II. vagy a III. fejezeteiben bemutatott szimulációs eredmények azt vetítik előre, hogy egyetlen célfüggvény korántsem elég a megfelelő eredmény eléréséhez.

Dim.	Kalkuláció		Célfüggvény					
	min.	max.	min.	max.	átlag	medián	std.dev.	std.err.
5	5049	11169	0,05	0,09	0,07	0,08	0,02	0,01
50	146652	479851	0,32	0,49	0,40	0,42	0,07	0,03
200	1260740	3609000	0,93	1,19	1,03	0,99	0,09	0,04

Dim.	Futási idő CPU*-val és GPU-val (sec)				Futási idő csak CPU*-val (sec)				Másodlagos
	min.	max.	átlag	medián	min.	max.	átlag	medián	célfüggvény
5	0,83	2,31	1,56	1,55	3,10	6,90	4,98	5,16	0
50	30,52	99,89	56,68	35,52	95,85	315,16	165,16	95,85	0
200	290,65	788,14	488,62	470,06	866,66	2469,64	1465,64	1444,73	0-8

7. táblázat. Új típusú FWA algoritmus statisztikai szimulációs paraméterei CPU* és GPU használatával (*15 core)

Az eredmények azt mutatják, hogy az általam javasolt új típusú FWA algoritmus jelentősen javított a HTC függvény predikciójának a teljesítményén, miközben az eredmények minősége egyáltalán

nem romlott a **módosított** PSO és FWA algoritmushoz képest (8. táblázat). Azért a módosított algoritmusokhoz hasonlítottam az új típusú FWA algoritmus eredményeit, mert a szakirodalomban található standard általam vizsgált PSO és FWA algoritmusok a dolgozatban vizsgált 5 dimenziós keresési teret kivéve alkalmatlanok a feladatra, értékelhetetlen eredményeket produkáltak.

Algoritmus	Dim.	Kalkuláció[a]	Célfüggvény[a]				Futási idő[a] (sec)	
		max.	stop	max.	median	std.dev.	max.	medián
PSO[b]	5	150050	0.1	851.52	13.20	329.06	31.06	15.49
QPSO[c]	5	150050	0.1	2428.13	2.70	997.25	30.49	15.42
FWA[d]	5	300200	0.1	1.22	0.32	0.43	63.16	60.75
nFWA	**5**	**11169**	0.1	**0.09**	**0.08**	**0.02**	**2.31**	**1.55**
PSO[b]	50	300200	0.5	0.95	0.27	0.32	65.28	64.67
QPSO[c]	50	300200	0.5	1.64	0.41	0.59	59.97	57.55
FWA[d]	50	600400	0.5	407.64	279.51	98.16	134.81	135.65
nFWA	**50**	**479851**	0.5	**0.49**	**0.42**	**0.07**	**99.89**	**35.52**
PSO[b]	200	1200400	1	265229.33	21841.88	103275.26	468.59	463.57
QPSO[c]	200	1200400	1	69.04	12.48	24.60	463.17	456.45
FWA[d]	200	3601800	1	221.07	108.40	43.90	883.59	859.81
nFWA	**200**	**3609000**	1	**1.19**	**0.99**	**0.09**	**788.14**	**470.06**

a: átlagolt értékek
b: átlagolt értékek algoritmusai: PSO, PSOIn, PSOCo
c: átlagolt értékek algoritmusai: QPSOT1, QPSOT2
d: átlagolt értékek algoritmusai: FWA, AFWA, CFWA, EFWA, EFWADM

8. táblázat. 7., M2. és M3. táblázatok eredményeinek átlagolt értékei

Átlagolva az eredményeket az újfajta FWA algoritmus nagyobb és robusztusabb teljesítményre képes mint az általam vizsgált másik 10 hasonló bio-inspirált eljárás, miközben a realisztikus HTC alakja nem tér el egyik irányba sem a vizsgált bio-inspirált eljárások eredményétől. A táblázatból az olvasható ki, hogy míg sebesség tekintetében az újfajta FWA algoritmussal (a táblázatban nfwa néven szerepel) a módosított PSO és FWA algoritmusokkal összemérhető futási időket mértem, de a módosított PSO és FWA algoritmusok általában nem tudják befejezni a számításokat a HTC függvény predikciójához a kitűzött kalkulációs szám elérése előtt. Az újfajta FWA algoritmus majdnem minden esetben elérte a kitűzött célfüggvényt a kitűzött kalkulációs szám elérése előtt, a számítások során elért célfüggvény tekintetében 1 nagyságrenddel jobb eredményeket mutatott. Az általam vizsgált bio-inspirált algoritmusokat minden esetben véletlen pozíciókból indítottam.

*6.11. **4. Tézis***

Kidolgoztam egy, a tranziens hőátadási együttható függvény becslésére, speciálisan az 1D tengelyszimmetrikus, a test felületi hőátadási függvényének becslésére alkalmas új típusú Fireworks algoritmust. Az új számítási eljárás optimalizálási korlátként veszi figyelembe az időben változó hőátadási függvény

predesztinált karakterisztikáját. A Fireworks algoritmus alapegységeit jelentő sparkok helyzeteinek meghatározásához új típusú mozgásegyenleteket javasoltam. Az új módszer számítási hatékonyságát 50 és 200 pontból álló hőátadási együttható függvények becslésének esetében, numerikus tesztek eredményei alapján igazoltam.

6.12. Tézishez kapcsolódó saját publikációk

1. Zoltán Fried, Sándor Szénási és Imre Felde: „Prediction of objective function value for heat transfer coefficient function reconstruction by FWA”. *2019 IEEE 13th International Symposium on Applied Computational Intelligence and Informatics (SACI)*. Timisoara, Romania: IEEE, 2019. máj., 305–308. old. ISBN: 978-1-7281-0685-4. DOI: 10.1109/SACI46893.2019.9111623.

7. A hőátadási együttható függvény becslésének egy egyszerűsített megoldása

Ebben a fejezetben olyan kutatást mutatok be, amely henger alakú munkadarab felületén feltételezett speciális $h(\vec{r}, t)$ eloszlás alkalmazása helyett feltételezi, hogy a HTC együttható a felületen közvetlenül a hőmérséklet függvénye, azaz egyszerűbb $h(T(\vec{r}, t))$ függvénnyel van dolgunk, továbbá kvalitatív megfontolások alapján a $h(T)$ függvény jellege leírható néhány „alakparaméterrel”. E függvény egyszerűen módosítható alakparaméterei „hangolásával”.

A fenti matematikai megszorítás nagyon jelentős egyszerűsítés (komplexitásredukció), amelynek eredményeként az optimumkeresési feladat algoritmusa kisebb teljesítményű számítógépen is alkalmazható.

A bio-inspirált módszereknek egyik nagy hátránya, hogy a keresett gyökök pontos értékét csak nagyon nagy energiák befektetése révén lehet meghatározni, vagyis túl sokáig tart a futási idő, ha egyáltalán sikerül a globális minimum közelébe kerülni. Viszont arra tökéletesen megfelelnek, hogy a lehetséges megoldás közelébe vigyék az algoritmust, és onnan már valamilyen gradiens vagy gradienshez hasonló módszerrel meg tudjuk határozni a pontos értékeket. Ez a kitétel fokozottan igaz magasabb dimenziójú keresési terek esetében.

A nagy számítási kapacitást felhasználó modellezés kiegészítéseként egészen egyszerű és kis számítási igényű „alternatív módszer” bevezethetőségét vizsgálom meg egy 1D tengelyszimmetrikus ötvözet felületén mérhető HTC hűlés közbeni időfüggésének meghatározásához.

7.1. A feladat végeselem közelítése hengerkoordináták használatával

Nyilvánvaló, hogy a hengerkoordináták központi tengelyben való szingularitása miatt az $r = 0$ középvonalban vett értékeket numerikusan nem tudjuk „pontosan” kezelni. A probléma elkerülése érdekében az „egzakt” teljes $[0, R]$ tartomány helyett a közelítő $[\Delta r, R]$ tartományt vizsgáltam, amelyet úgy kaptam, hogy a teljes $[0, R]$ intervallumot $\breve{N} \in \mathbb{N}$ darab egyenlő, $\Delta r = \frac{R}{\breve{N}}$ hosszúságú szakaszra osztottam fel, és a középvonal szerepében az $r = \Delta r$ értéket használtam. Az $\{r_i | i = 2, \ldots, \breve{N} - 1\}$ rácspontok esetében a gradiens *centrális közelítést* alkalmaztam az (57a) szerint. Ebből eredően az (5) egyenletben szereplő $\nabla(k \cdot \nabla T)$ gradienst csak az $\{r_i | i = 3, \ldots, \breve{N} - 2\}$ pontokra lehetett így kiszámítani, míg a „középvonalra” az (57c) közelítést használom.

A hőmérséklet értékek idő szerinti frissítésére $\frac{\partial T}{\partial t}$ alapján így csak az $\{r_i | i = 3, \ldots, \breve{N} - 2\}$ pontok

esetében volt mód.

$$\frac{\partial T(r_i,t)}{\partial r_i} \approx \frac{T(r_{i+1},t)-T(r_{i-1},t)}{r_{i+1}-r_{i-1}} \qquad i\in\{2,\dots,\check{N}-1\} \tag{57a}$$

$$\frac{\partial}{\partial r}\left(k\frac{\partial T}{\partial r}\right) \approx \frac{k(r_{i+1},t)\nabla T(r_{i+1},t)-k(r_{i+1},t)\nabla T(r_{i-1},t)}{r_{i+1}-r_{i-1}} \qquad i\in\{3,\dots,\check{N}-2\} \tag{57b}$$

$$T(r_1,t)\equiv T(r_2,t)\equiv T(r_3,t) \tag{57c}$$

A (6) egyenletben adott határfeltétel teljesítése érdekében az $r_{\check{N}-1}$ rácspontban frissített T értéke a már frissített többi pont 1. térváltozó szerinti deriváltjának használatával becsülhető az (58) szerint.

$$T(r_{\check{N}-1},t) \approx T(r_{\check{N}-2},t) + \frac{(r_{\check{N}-1}-r_{\check{N}-2})(T(r_{\check{N}-2},t)-T(r_{\check{N}-3},t))}{(r_{\check{N}-2}-r_{\check{N}-3})} \ , \tag{58}$$

és $T(r_{\check{N}},t)$ értéke a hőátadási tényező segítségével szintén becsülhető az (59) szerint:

$$T(r_{\check{N}},t) \approx \frac{h(T(r_{\check{N}-1},t))T_q + k(T(r_{\check{N}-1},t))/\Delta r}{h(T(r_{\check{N}-1},t)) + k(T(r_{\check{N}-1},t))/\Delta r} \ . \tag{59}$$

7.2. A numerikus számítások és a végeselem közelítés paramétereinek beállítása

A szimulációkban a $h(T)$ függvény becslésére kiindulásként a (60) egyenletben adott függvényformát választottam, amely tipikus aszimmetriát mutat az „alacsonyabb" és a „magasabb" hőmérséklet-tartományban mutatott viselkedésében:

$$h(T) = h_{\max}\cdot\begin{cases}\exp\left(-\left(\left[T-T_{\max}\right]/w_{\text{left}}\right)^{\tilde{p}}\right), & ha\ T\le T_{\max}\\ \exp\left(-\left(\left[T-T_{\max}\right]/w_{\text{right}}\right)^{\tilde{p}}\right), & ha\ T>T_{\max}\end{cases} \tag{60}$$

melyben $h_{\max}$ a feltételezett maximális HTC értéket, $T_{\max}$ pedig azt a hőmérsékletet jelöli, amely a $h_{\max}$ érték „helye", a w_{left} és w_{right} „szélességparaméterek" pedig az aszimmetrikus eloszlások egy-egy oldalához tartoznak. A $\tilde{p}$ paraméter a függvény „farokrészének" lecsengését határozza meg, ennek értékét állandónak tekintem a gradiens kiszámításánál.

A fenti függvényformát plauzibilis fizikai megfontolások alapján választottam, amelyek ugyanúgy érvényesek a hűtőgépek működésében, mint a fémek hűtésében, csak az előbbi esetben sokkal alacsonyabb hőmérséklet-tartományban játszódnak le. Ez az alacsonyabb hőmérséklet lehetővé teszi a jelenség optikai megfigyelését átlátszó üvegcsövek alkalmazásával ([113–115]). A HTC érték egy mérésekkel jól megfigyelhető paraméter, ami mögött komplikált fizikai folyamatok rejlenek:

a) A hűtőfolyadék áramlása általában turbulens, amely nagyon hatékony hőátadást tesz lehetővé az egymással összekeveredő „folyadékrétegek" között, kivéve a munkadarab felületén kialakuló „felületi réteget". Ebben, mivel a folyadék „tapad" a munkadarab felületén, lamináris áramlás

valósul meg, ami mérsékelt hőátadási képességet eredményez az összekeveredett „bulk” folyadéktömb felé. Ebben a rétegben a hőátadás főleg hővezetéssel történik, mert annak iránya merőleges az áramlás irányára, ez pedig nem egy hatékony hőátadási forma. A jelenség erősen függ a tekintett felület minőségétől, „érdességétől”.

b) A folyadék viszkozitása általában csökken a hőmérséklet növekedésével, ami miatt magasabb hőmérsékleteken vékonyabb a felületi réteg kialakulása, így a „bulk” folyadékanyag felé hatékonyabb hőátadás várható.

c) Amikor a hőmérséklet eléri a folyadék forráspontját az adott nyomáson, a munkadarab felületén gázbuborékok képződnek, amelyek hőszigetelőként működve csökkentik a HTC értékét. Hasonlóan működik a hűtőfolyadékban oldott, a légkörből származó természetes gázok kiválása is, ami általában a forráspontnál alacsonyabb hőmérsékleten már elkezdődik.

A fenti folyamatok „precíz modellezése” igen nehéz és szinte reménytelen, viszont azok alátámasztják a (60) egyenletben használt kvalitatív formát. Ennek alapján remélhető, hogy a benne adott 4 paraméter értékének illesztésével a jelenség közelítő modellezése megvalósítható.

Igen sok „komplikált” függvényforma jól közelíthető „egyszerű”, numerikusan jól kezelhető képletekkel. Például a „normális”, az „epsilon”, és az „omega” eloszlás függvényének leírására Dombi egyszerű paraméteres alakot javasolt ([116–118]). Ugyancsak Dombi tett javaslatot a „kappa regressziós függvény” egyszerű közelítésére [119]-ben. A gyakorlatban könnyen lehet kvalitatíve elfogadható közelítést kapni a 8/b ábrán látható görbékre a (60) egyenletben adott képlet használatával a 9. táblázatban adott célparaméterekkel.

Paraméter	**Cél**	**Kezdeti**
$h_{\max}$ $\left[\frac{W}{m^2 \cdot K}\right]$ maximális HTC	5700,0	5500,0
$T_{\max}$ $[K]$ a maximális HTC helye	680,0	650,0
w_{left} $[K]$ bal oldali szélesség	260,0	300,0
w_{right} $[K]$ jobb oldali szélesség	80,0	60,0
p [dimenziótlan] „farok lecsengés” paraméter (rögzített)	2,0	2,0

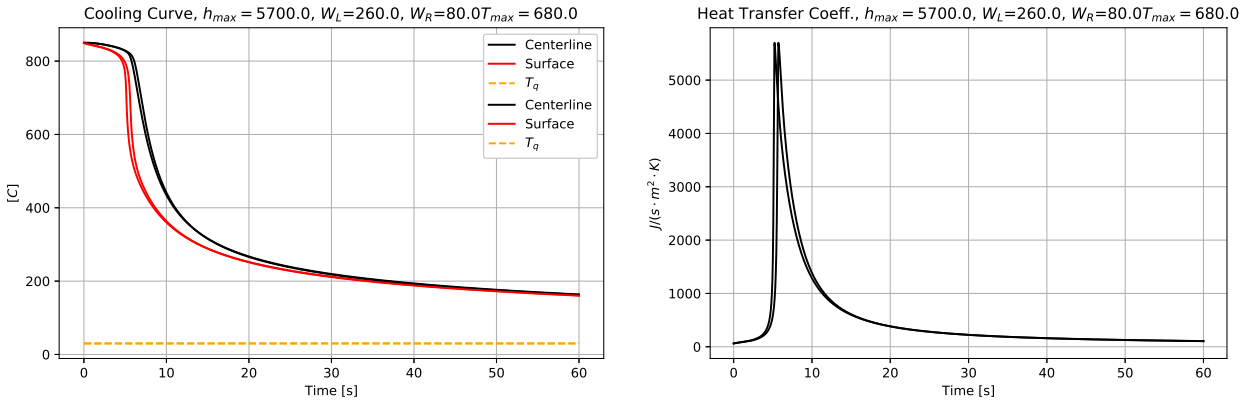
9. táblázat. A „célparaméterek” és a „kezdeti modellparaméterek” a (60)-ban

A 35. ábrán a 7.1. alfejezetben részletezett két „közelítő numerikus paraméterpárra” ($\{\check{N} = 50,\ \delta t = 10^{-3}\ [s]\}$ és $\{\check{N} = 20,\ \delta t = 10^{-2}\ [s]\}$) kapott eredményeket egy közös grafikonra nyomtattam a könnyebb összehasonlíthatóság érdekében. A „célparaméterek” a 9. táblázatban adottak. (Az idő szerint egyszerű Euler-féle integrálást alkalmaztam a megfelelő δt időfelbontással.) Látható, hogy a sok számítást igénylő „finom felbontás” majdnem pontosan arra az eredményre vezet, mint a jóval kisebb számítási igényű „durva felbontás”. Emiatt a további számításokat ez utóbbi, azaz az $\{\check{N} = 20,\ \delta t =$

$= 10^{-2}\ [s]\}$ beállításokkal végeztem. (A 8/b ábrán a különböző közelítési módszerekhez eléggé eltérő eredmények tartoznak.)

7.3. Szimulációs eredmények

A fentiekben részletezett közelítésekkel az adott $T_q = 30{,}0\ [°C]$ hűtőfolyadék hőmérséklettel, $T_{ini} = 850{,}0\ [°C]$ kezdeti munkadarab-hőmérséklettel a $\delta t \cdot [1, 6000]$ időrácson egy többváltozós függvény konstruálható egy 4 *dimenziós bemeneti térrel*, ami a 9. táblázat változtatható paramétereinek felel meg. A hibát mint egy sokváltozós tömbhöz rendelhető Frobenius-féle normát vezettem be mint „a célhőmérséklet az idő függvényében" és a „szimulált hőmérséklet az idő függvényében" tömbök különbségére nézve: $E(x) : \mathbb{R}^4 \mapsto \mathbb{R}$. Ennek minimumát kellett megtalálni a $h(t)$ függvény megkereséséhez.



35. ábra. A „hűlési görbe" és a „HTC az idő függvényében" függvények, amelyek a (60) egyenlethez tartoznak

A „Klasszikus Mechanika" Lagrange-féle leírására a Lagrange által 1811-ben bevezetett „Redukált Gradiens Módszer" [120] implementálása lassú algoritmusra vezet még a kényszeregyenletek hiánya esetében is, ha nem rendelkezünk előzetes információval $E(x)$ várható minimumára vonatkozóan. Ilyenkor egy lokális helyi minimumba konvergál a lassú algoritmus. Ezzel szemben például a klasszikus Newton-Raphson algoritmus ([121]) gyors eredményre tud vezetni, ha a feltételezett $min\ E(c) = 0$ állítás valóban igaz. Az alapvető megfontolások ebben az algoritmusban az alábbiak: válasszunk ki egy $x(1)$ „kezdeti pontot", számítsuk ki $\nabla E(x(1))$ értékét, és tegyünk egy $\Delta x = \tilde{\alpha} \nabla E(x(1))$ „nagy ugrást" úgy, hogy teljesüljön a $-E(x(1)) = \Delta x^T \nabla E(x(1))$ feltétel. Ha a tekintett függvény lineáris vagy affin szerkezetű lenne, ez az egyetlen ugrás belevezetne a lokális minimumba. Ha a függvényünk nemlineáris, ismételjük meg ezt a lépést az új, $x(2) = x(1) + \Delta x$ pontból kiindulva, és így tovább. A

lépések kiválasztása a (61) egyenlet szerint történik.

$$\tilde{\alpha} = \frac{-E(x(1))}{\|\nabla E(x(1))\|^2} \tag{61}$$

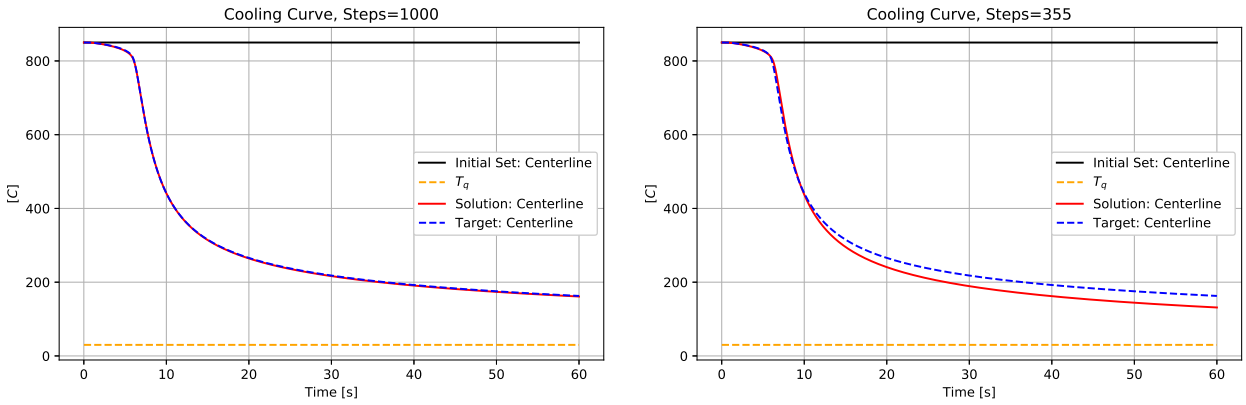
Tapasztalatból tudom, hogyha a ∇E gradienst nem precízen számolom, az algoritmus könnyen divergálhat, továbbá ha a *min* $E(c) = 0$ állítás nem pontosan igaz, a megoldás „ugrál” a minimum helye körül. Emiatt az eredeti Newton-Raphson algoritmust módosítottam a következő fejezetben.

7.3.1. Egzakt megoldással rendelkező feladat esete

Az eredeti, a Newton-Raphson algoritmusban használt a (61) egyenletben adott lépést a (62) szerint módosítottam, hogy a feladat megoldásában ne okozzon zavart:

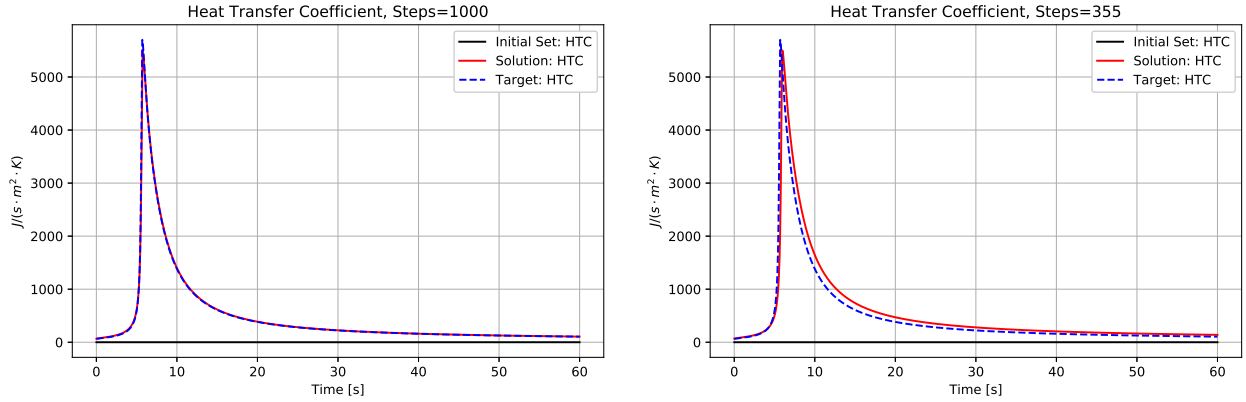
$$\tilde{\alpha} = \gamma \frac{-E(x(1))}{\|\nabla E(x(1))\|^2 + \varepsilon} \ , \tag{62}$$

melyben a $\varepsilon = 10^{-14}$ paramétert a 0-val való osztás kiküszöbölésére vezettem be, a „finomító faktor” $\gamma = 5 \cdot 10^{-3}$ értékét pedig tapasztalatilag állítottam be szimulációk futtatásával. A $\nabla E(x)$ gradiens értékét numerikusan becsültem a 9. táblázatban definiált paraméterek „aktuális értékétől” való kis elmozdításával: $h_{\max} \longrightarrow h_{\max} + \Delta h_{\max}$, $T_{\max} \longrightarrow T_{\max} + \Delta T_{\max}$, $w_{\text{left}} \longrightarrow w_{\text{left}} + \Delta w_{\text{left}}$, és $w_{\text{right}} \longrightarrow w_{\text{right}} +$ $+ \Delta w_{\text{right}}$ a következő „eltolás” értékekkel: $\Delta h_{\max} = 10^{-3} \left[\frac{W}{m^2 \cdot K}\right]$, $\Delta T_{\max} = 10^{-3}\ [K]$, $\Delta w_{\text{left}} =$ $= 10^{-3}\ [K]$, valamint $\Delta w_{\text{right}} = 10^{-3}\ [K]$. A gradienst a szomszédos értékekre kapott eltérésekből számoltam. A γ paramétert a kiindulási 1 értékről addig csökkentettem, amíg a nagy végső hibák közti „ugrálás” az eredményekben megszűnt. A kapott megoldás egyféle „átmenet” a Lagrange-féle Gradiens Módszer és a Newton-Raphson algoritmus között, amely a pontosság és a futási idő közti kompromisszum alapján volt beállítható.

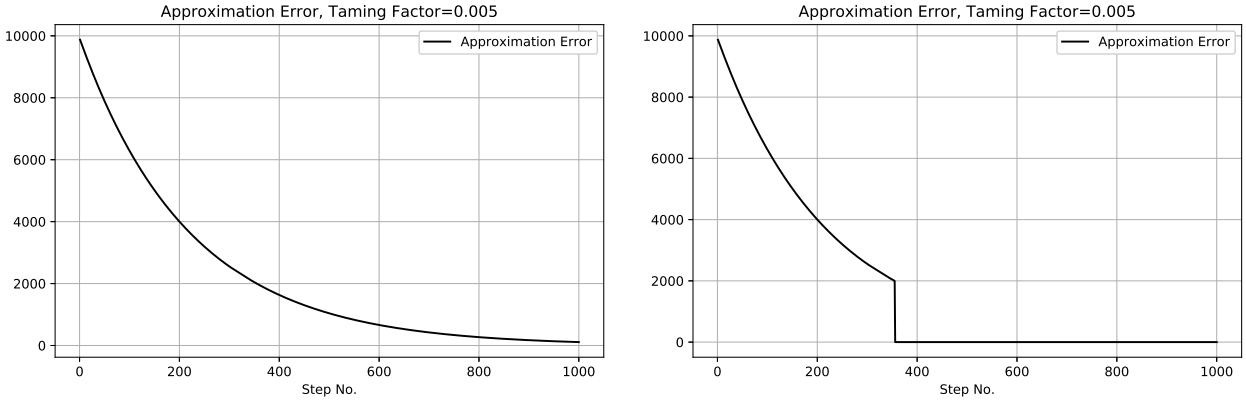


36. ábra. A módosított Newton-Raphson algoritmus működése az egzakt megoldással bíró feladat esetén: a „hűlési görbék” 1000 lépésre számolva (balra), és korlátozott lépésszámra számolva, mikor az algoritmus leállt az $E = 2000$ hibaérték elérése után (jobbra)

A 36.–38. ábrák tanúsága szerint a szimuláció konzisztens és értelmezhető eredményeket szolgáltatott az egzakt megoldással bíró feladat esetében. Itt is a (60) egyenletben a 9. táblázatban adott paramétereket használtam.



37. ábra. A módosított Newton-Raphson algoritmus működése az egzakt megoldással bíró feladat esetén: a hőátadási tényező („$h(t)$ függvény”) 1000 lépésre számolva (balra), és korlátozott lépésszámra számolva, mikor az algoritmus leállt az $E = 2000$ hibaérték elérése után (jobbra)



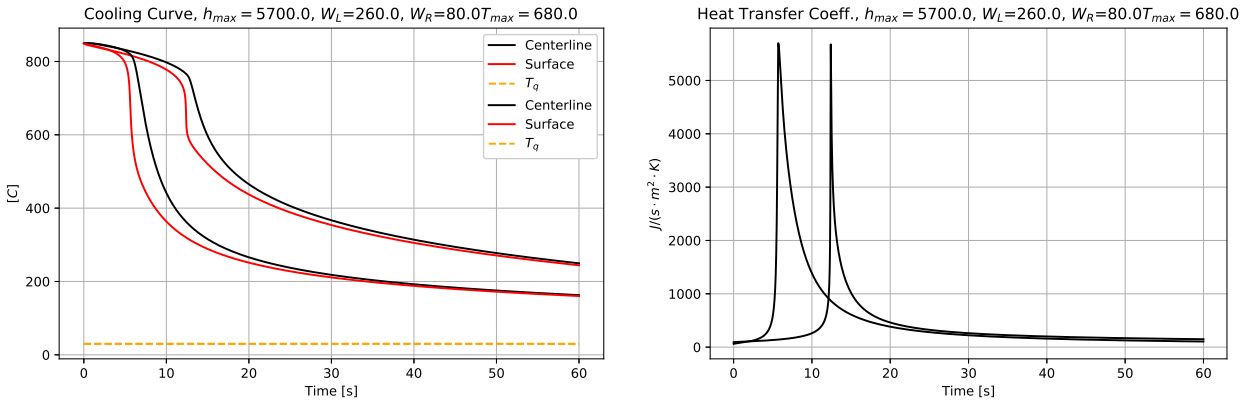
38. ábra. A módosított Newton-Raphson algoritmus működése az egzakt megoldással bíró feladat esetén: a közelítés hibája („$F(t)$ függvény”) 1000 lépésre számolva (balra), és korlátozott lépésszámra számolva, mikor az algoritmus leállt az $E = 2000$ hibaérték elérése után (jobbra)

7.3.2. Egzakt megoldással nem rendelkező feladat esete

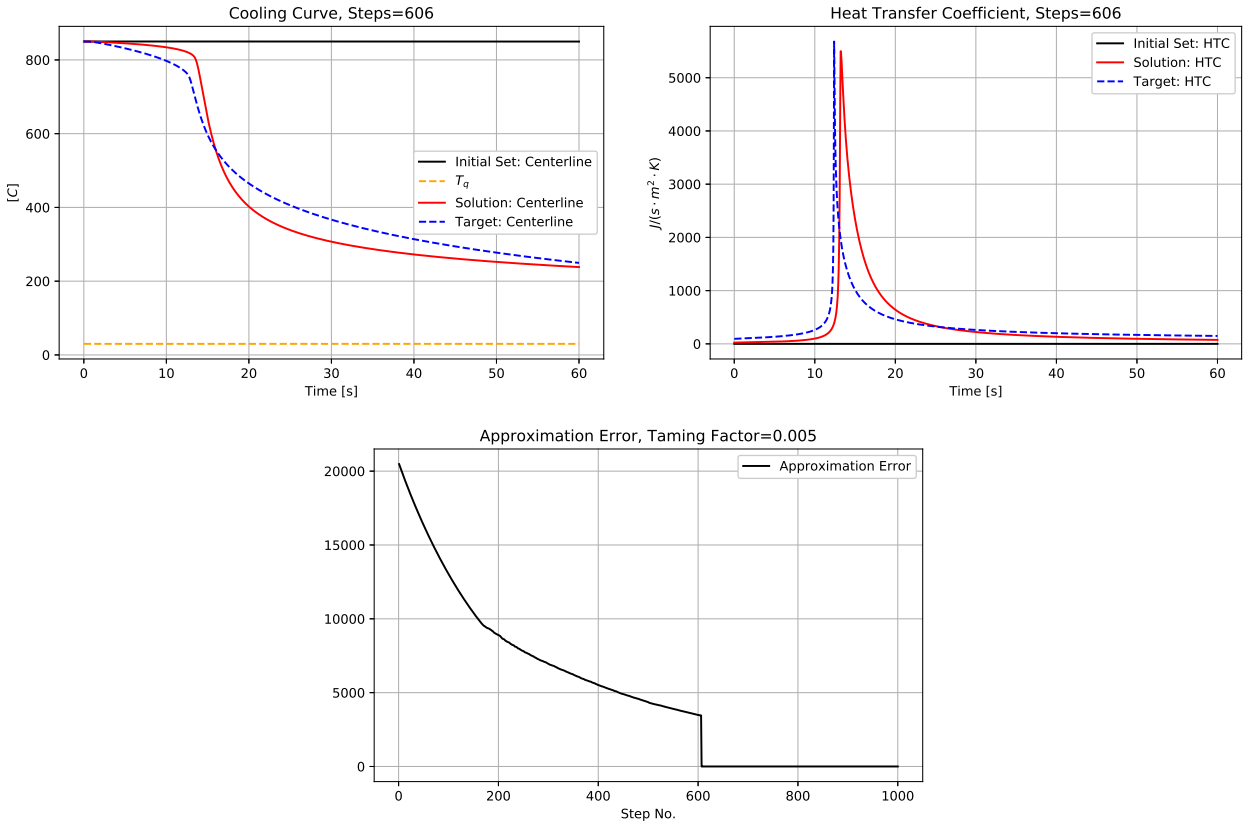
Annak érdekében, hogy a javasolt módszert pontos megoldással nem rendelkező feladatra is teszteljem, olyan „céleloszlást” vezettem be, amelyet a használt közelítő függvénytől különböző függvényformával állítottam elő a (63) egyenlet szerint a $\tilde{d} = 7.5 \cdot 10^{-2}$ „deformációs paraméter” használatával.

$$h(T) = h_{\max} \cdot \begin{cases} \dfrac{\tilde{d}}{\tilde{d}+([T-T_{\max}]/w_{\text{left}})^{\bar{p}}}, & ha\ T \leq T_{\max} \\ \dfrac{\tilde{d}}{\tilde{d}+\left([T-T_{\max}]/w_{\text{right}}\right)^{\bar{p}}}, & ha\ T > T_{\max} \end{cases} \tag{63}$$

A „céleloszlás” módosítása a 39. ábrán látható eredményt adta. Látható, hogy mindkét céleloszlással hasonló jellegű görbéket lehetett nyerni.



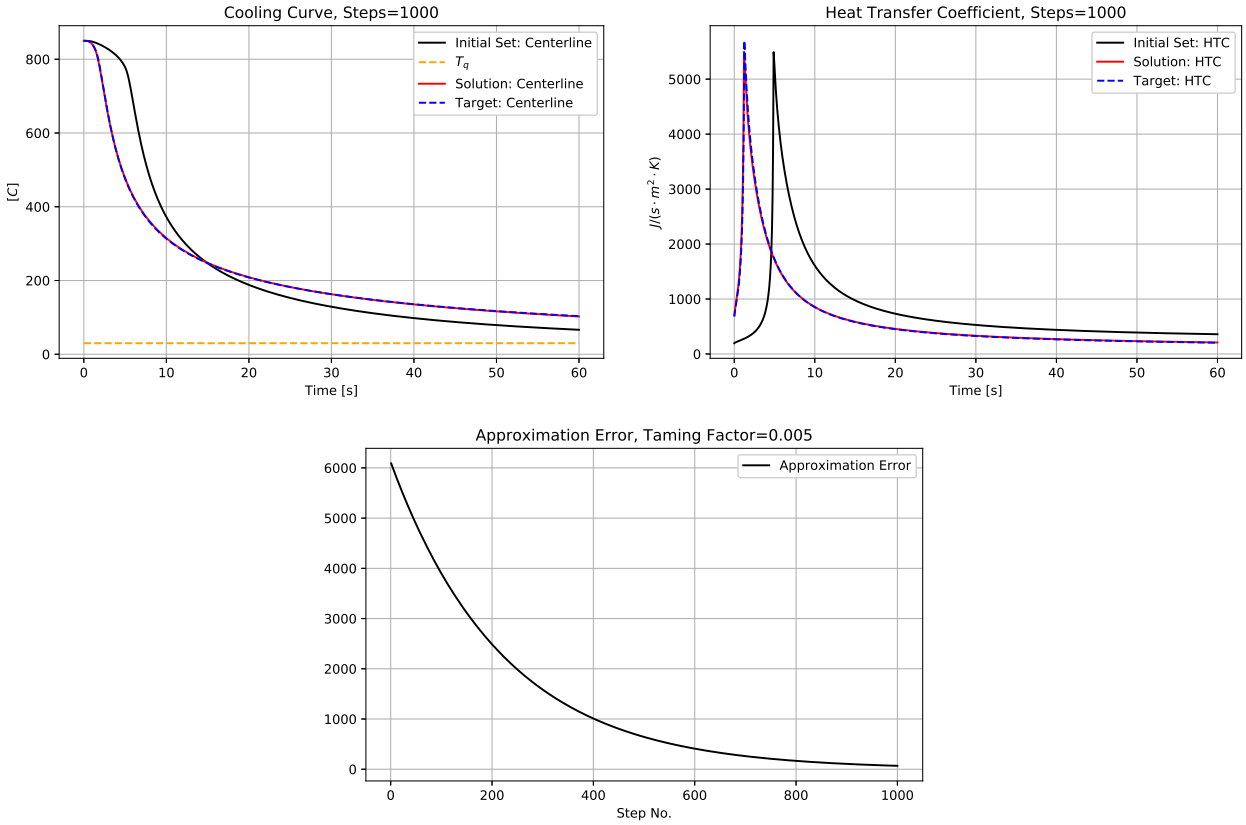
39. ábra. A „deformációs paraméter” $\tilde{d} = 7{,}5 \cdot 10^{-2}$ értékének hatása a céleloszlás módosulására a (63) szerinti generálásával (balra) és a hozzájuk tartozó, „egzakt” megoldásként megtalált $h(t)$ függvények (jobbra) (a megfelelő görbéket közös grafikonon ábrázoltam)



40. ábra. A módosított Newton-Raphson algoritmus működése egzakt megoldással nem bíró feladat esetében

A 40. ábra jól mutatja, hogy bár „egzakt megoldást” nem lehetett találni, az algoritmus jól megközelítette a „deformált céleloszlást”, és sikerrel konstruált hozzá egy lehetséges, realisztikusnak

tűnő $h(t)$ függvényt. Az algoritmus megállt amikor a hiba további csökkentése már nem volt lehetséges. A legkisebb megengedett hiba értékét 1,0-ban, a maximális lépésszámot pedig 1000-ben határoztam meg.



41. ábra. A módosított Newton-Raphson algoritmus működése az egzaktul megoldható feladatra

7.4. Az aszimmetria további növelése a komplexitás csekély mértékű növelésével

Még realisztikusabb függvényalakok előállításához egyszerű lehetőség a (60) egyenletben adott forma további módosítása különböző lecsengést biztosító kitevő bevezetése a $h(T)$ függvény „bal” ($\tilde{p}_{\text{left}}$) és „jobb” ($\tilde{p}_{\text{right}}$) oldalán a (64) és a (65) egyenletek szerint:

$$h(T) = h_{\max} \cdot \begin{cases} \exp\left(-\left(\left[T - T_{\max}\right] / w_{\text{left}}\right)^{\tilde{p}_{\text{left}}}\right), & ha\ T \leq T_{\max} \\ \exp\left(-\left(\left[T - T_{\max}\right] / w_{\text{right}}\right)^{\tilde{p}_{\text{right}}}\right), & ha\ T > T_{\max} \end{cases} \tag{64}$$

és

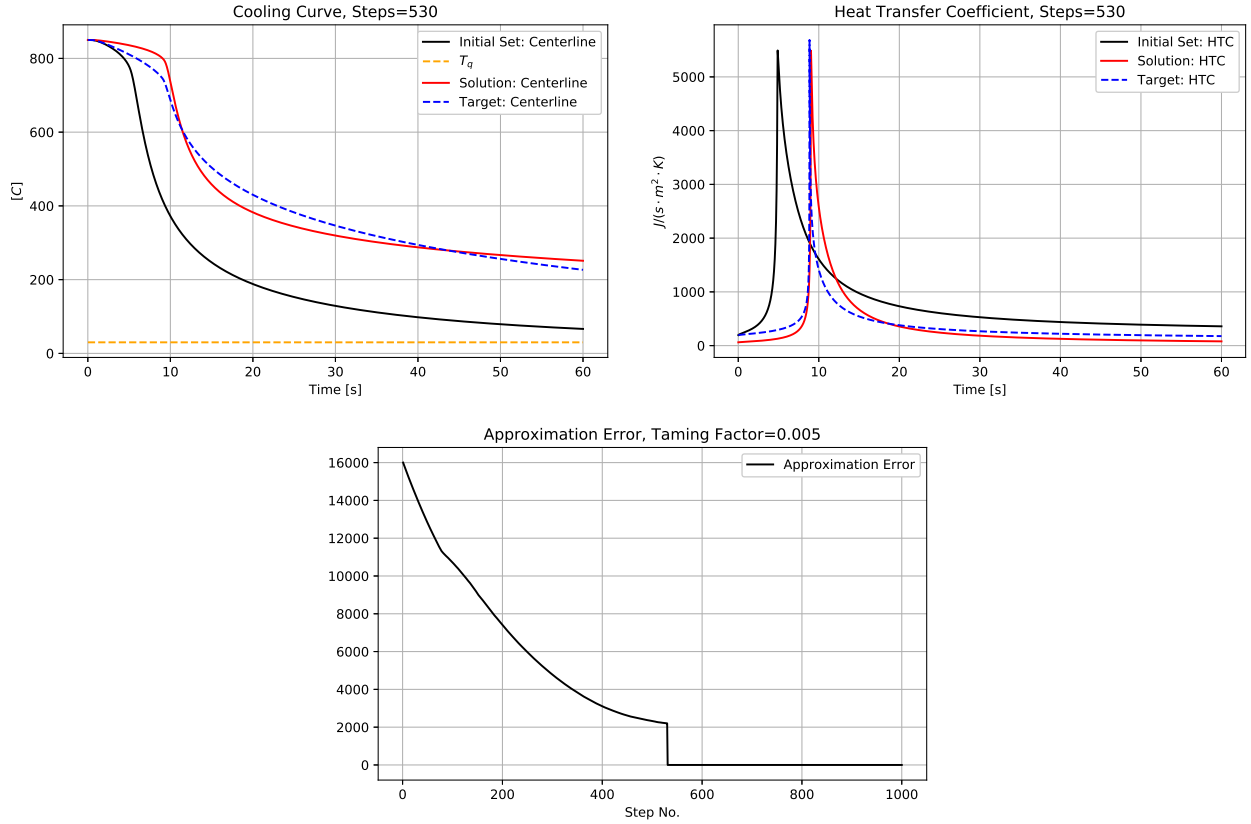
$$h(T) = h_{\max} \cdot \begin{cases} \frac{\tilde{d}}{\tilde{d}+\left(\left[T-T_{\max}\right]/w_{\text{left}}\right)^{\tilde{p}_{\text{left}}}}, & ha\ T \leq T_{\max} \\ \frac{\tilde{d}}{\tilde{d}+\left(\left[T-T_{\max}\right]/w_{\text{right}}\right)^{\tilde{p}_{\text{right}}}}, & ha\ T > T_{\max} \end{cases} \tag{65}$$

E paraméterek további hangolása a bemeneti tér dimenzióját mindössze 2-vel növelné, ami nem jelent drasztikus dimenziószám növekedést a gradiens alapú módszer használatában.

7.5. Eredmények értékelése

A 41. ábra tanúsága szerint az egzaktul megközelíthető megoldást kapjuk meg a (64) egyenletben rögzített $\tilde{p}_{\text{left}} = 1{,}5$ és $\tilde{p}_{\text{right}} = 1{,}0$ paraméterekre. Az alsó hibahatár és a maximális lépésszám értéke 1,0 és 1000, és az algoritmus megállt, amikor a hiba további csökkentése lehetetlen volt.

A 42. ábra a (65)-ben adott alak a (64)-ben adott alakkal való közelítéséhez tartozik, amelynek nem lehet „egzaktul pontos” eredménye. A rögzített paraméterek szintén a $\tilde{p}_{\text{left}} = 1{,}5$ és $\tilde{p}_{\text{right}} = 1{,}0$ voltak, az alsó hibahatár és a maximális lépésszám értéke szintén 1,0 és 1000. Az algoritmus megállt, amikor a hiba további csökkentése a továbbiakban lehetetlennek bizonyult. A javasolt módszer ebben az esetben is ígéretes eredményt adott.



42. ábra. A módosított Newton-Raphson algoritmus működése egzaktul nem megoldható feladat esetén

A javasolt megoldás hatékonyságának mérésére a program futási ideje egy mérvadó paraméter. A 40. ábrán kapott eredmények a 2.13. szakaszban specifikált hardver és szoftver környezetben 6 perc 48 másodperc futási idő alatt készültek el, ami a választott programnyelv és a párhuzamosíthatóság kihasználatlanságát figyelembe véve nagyon jó eredmény. Hozzá kell tenni, hogy ez a futási idő csupán, további programozástechnikai eljárásokkal (párhuzamosítás, GPU és optimalizált memóriahasználat) tovább csökkenthető. A kutatásaim eredményei az alábbi tézisben foglalhatók

össze:

7.6. ***5. Tézis***

Egyszerű és kis számítási igényű módszert javasoltam arra, hogy komplexitásredukciók alkalmazásával a mért „hűlési görbék” ismeretében megbecsülhető legyen a hőátadási együttható függvény időtől való függése az 1D tengelyszimmetrikus test középvonalában egy a klasszikus Gradiens Módszer és a szintén klasszikus „Newton-Raphson algoritmus” közti numerikus megoldáson alapuló eljárás segítségével közönséges kis-kapacitású számítógépen is.

7.7. Tézishez kapcsolódó saját publikációk

1. Zoltán Fried, Imre Felde és József K. Tar: „On the Simulation of Cooling Curves Using Simple Functional Formats”. *Acta Polytechnica Hungarica* 17.9 (2020), 109–124. old. ISSN: 1785-8860. DOI: 10.12700/aph.17.9.2020.9.6.

8. Összefoglalás

A dolgozatban új típusú számítási módszereket vezettem be a HTC függvény becslésére. Az általam bemutatott módszerek különböző, de már ismert számítási módszereken alapulnak (neurális háló, bio-inspirált algoritmusok, gradiens módszer). A bemutatott eljárásokkal az inverz hőtani probléma megoldásához szükséges számítási idő szignifikáns csökkentése érhető el, mivel az eljárások a HTC függvény becslésére optimalizáltak.

A módszerek alapját az a hipotézis képezi, hogy a nagy erőforrást igénylő matematikai modellek, műveletek egyszerűbben elvégezhető, de hasonló pontosságú eredményt adó modellekkel, műveletekkel helyettesíthetőek; illetve a repetitív, de egymástól független számítási műveletek párhuzamosíthatóak, miközben kihasználjuk, hogy a kapott eredmények pontossága vezérelhető - akár a számítások közben is, mellyel a számítási idő befolyásolható.

A kidolgozott eljárások többsége épít arra, hogy információval rendelkezik a számítások kezdetétől elért eredmények „megfelelőségének" mértékéről, amely lehetőséget teremt a felesleges számítások elkerülésére.

Az eljárásokkal 1D tengelyszimmetrikus (hengeres), végtelen hosszúságú próbatesten harmadfajú peremfeltétel mellett, radiális irányú hőközlést feltételezve, szimulációs modellek segítségével becsültem a felületi HTC függvényét. A modellek képességeit hipotetikus (mesterségesen generált) HTC függvények felhasználásával teszteltem. Az eredmények grafikus és táblázatos formája is igazolja a módszerek működőképességét.

Ezen vizsgálatok alapján arra következtetek, hogy a bevezetett új modellek a hagyományos eljárásokkal kombinálva gyorsabb, és egyben hasonló pontosságú eredmények elérésére képesek, mint a hagyományos eljárások.

Az értekezés új tudományos eredményei a következőképpen foglalható össze:

1. Az FWA és PSO algoritmusok esetében az algoritmusokat kiegészítettem két általam fejlesztett megoldással (másodlagos célfüggvény, mapping operátor), amiknek hatására, függetlenül az FWA és PSO algoritmusok pontos hangolási paramétereitől, drasztikusan csökkenthető a számításokhoz szükséges futási idő.
2. Az új típusú FWA algoritmus esetében újfajta mozgási egyenlettel rendelkező részecskét mutatok be, amivel a HTC függvény becslése esetén a keresési tér bejárásához szükséges lépések nagyságát lehet adaptív módon szabályozni.
3. Egy csökkentett számítási igényű, gradiens módszert mutatok be a HTC függvény becslésére,

amelyek lényege egyfajta komplexitásredukció. Ez abból áll, hogy az általános $h(\mathbf{r}, t)$ függvény helyett a $h(T)$ egyváltozós függvényre vezetek be egyszerű paraméteres formákat.

4. Neurális hálózat felhasználásával mutatok be egy gépi tanulási modellt, amellyel a test középvonalában kialakult hőmérsékleti görbe alapján a felületi HTC függvény becsülhető.
5. A lehűlési görbék meghatározásához szükséges hőegyenlet számítást speciálisan GPU-ra optimalizáltam.

Az edzés során végbemenő nedvesítési front mozgása szélsőséges esetben a munkadarab deformációját vagy károsodását is okozhatja. A folyadék nedvesítési kinetikája a felületi HTC függvénnyel jellemezhető. A hőkezelés eredményeként kialakuló tulajdonságok szempontjából az edzés eredményességét nagyrészt a célszerűen megválasztott hűtőközeg dönti el. Amennyiben megfelelő pontossággal becsülhető a felületi HTC függvény, a munkadarab edzés utáni tulajdonságai pontosan becsülhetővé válik.

Vasúti sínek várható (pontos) élettartalmának becslése (esetleges törés szempontjából) a biztonságos „és olcsóbb” vonatozás feltétele. Jelenleg a lefektetett vasúti sínek cseréjének pontos idejét előzetes mérésekkel lehet csak meghatározni. A szükséges nagy számú mérések jelentős költsége, és időszükséglete okán azoknak csak töredékét végzik el, ami miatt a csere idejét csak tág határok között lehet meghatározni, vagyis a lefektetett vasúti síneket biztonsági okokból jóval az élettartamuk lejárta előtt (esetleges törés valószínűsége miatt) szükséges cserélni.

Nem edzés esetében, az élelmiszerek tartós tárolása illetve hűtése egy bevett eljárás, ahol az élelmiszer hűtésének és felengedésének folyamata alapvetően befolyásolja az élelmiszer felengedése utáni állagát, frissességét.

További kutatási irányok, gyakorlati alkalmazhatóság

1. Miért jelentős az eltérés a referenciagörbe csúcspontjától jobb oldalon található pontokhoz képest az FWA és PSO bio-inspirált heurisztikus eljárásokkal számolt görbe ugyanazon adataiban.
2. A 4. fejezetben ismertetett neurális háló becsléseként kapott HTC függvény felhasználásának lehetőségét a heurisztikus eljárások kezdeti állapotaként.
3. A dolgozatban ismertetett eljárások átültetése $2D$ tengelyszimmetrikus (hengeres), véges hosszú munkadarab esetére.
4. A heurisztikus algoritmusok számítási idejének legnagyobb hányadát a lehűlési görbék számítása teszi ki. Ennek a számítási időnek a további csökkentésével jelentősen csökkenteni lehetne a szimuláció futásának idejét.

9. Szoftver verziók és fontosabb paramétereik

- g++ verzió:

```
g++ (Ubuntu 9.4.0-1ubuntu1~20.04.2) 9.4.0
Copyright (C) 2019 Free Software Foundation, Inc.
```

- g++ fordítási paraméterek:

```
-O3 -mfma -mavx2 -std=c++11 -Wconversion -Wsign-conversion
-Wfloat-conversion -Wcast-align=strict -Wcast-function-type
-Wall -Werror -Wpedantic -Wstrict-aliasing
-Wno-deprecated-declarations -Wclobbered -Wcast-function-type
-Wdeprecated-copy -Wempty-body -Wignored-qualifiers
-Wimplicit-fallthrough=3 -Wmissing-field-initializers
-Wsign-compare -Wredundant-move -Wtype-limits -Wuninitialized
-Wshift-negative-value -Wunused-parameter
-Wunused-but-set-parameter
```

- nvcc verzió:

```
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2022 NVIDIA Corporation
Built on Wed_Sep_21_10:33:58_PDT_2022
Cuda compilation tools, release 11.8, V11.8.89
Build cuda_11.8.r11.8/compiler.31833905_0
```

- nvcc fordítási paraméterek:

```
-std=c++11 -I /usr/local/cuda/include/ -arch=sm_61 -lcudadevrt
```

- cuda verzió: 11.8
- python verzió: 3.8.8
- julia verzió: 1.8.3
- tensorflow verzió: 2.12.0
- ubuntu operációs rendszer (64bit) verzió: 20.04

10. Hardver elemek fontosabb paraméterei

Meghatározás	Paraméter
Tipus	Dell XPS 15
CPU	11. táblázat
Memória	Alder Lake PCH Shared SRAM, 16GB
Diszk	SSD, 2TB

10. táblázat. Laptop fontosabb paraméterei

Meghatározás	Paraméter
Processzorok száma	16
Gyártó	Intel
model name	11th Gen Intel(R) Core(TM) i7-11850H
cache size	24576 KB
flags	fpu mmx sse sse2 sse3 sdbg fma sse4_1 sse4_2 avx avx2 ...

11. táblázat. CPU fontosabb paraméterei

Meghatározás	Paraméter
CUDA Capability Major/Minor version number	7.5
Total amount of global memory	3914 MBytes (4104323072 bytes)
(16) Multiprocessors, (64) CUDA Cores/MP	1024 CUDA Cores
L2 Cache Size	1048576 bytes
Total amount of constant memory	65536 bytes
Total amount of shared memory per block	49152 bytes
Total number of registers available per block	65536
Warp size	32
Maximum number of threads per multiprocessor	1024
Maximum number of threads per block	1024
Max dimension size of a thread block (x,y,z)	(1024, 1024, 64)

12. táblázat. Nvidia T1200 Laptop GPU grafikus gyorsítókártya fontosabb paraméterei

11. Jelölések jegyzéke

$f(x)$	Célfüggvény
S	Célfüggvény értéke
S_{shape}	Alakra vonatkozó célfüggvény értéke
S_{best}	Célfüggvény aktuális optimális értéke
$\mathbf{S}$	Célfüggvény értelmezési tartománya
D	A keresési tér dimenziószáma
U	Felület, a hőáramlás keresztmetszete $\left[m^2\right]$
T	Hőmérséklet $[^\circ C]$
h	Hőátadási együttható (htc) $\left[\frac{W}{m^2 \cdot K}\right]$
$h_{\text{max}}, h_{\text{min}}$	Hőátadási együttható maximális és minimális értéke
λ	Hővezetési tényező $\left[\frac{J}{s \cdot m \cdot K}\right]$
ρ	Sűrűség $\left[\frac{kg}{m^3}\right]$
h_{max}	Maximális HTC érték $\left[\frac{W}{m^2 \cdot K}\right]$
C_p	Fajhő $\left[\frac{J}{kg \cdot ^\circ C}\right]$
Q	Hőmennyiség $[J]$
q_v	A látens hőmennyiség $\left[\frac{J}{m^3}\right]$
t	Idő $[s]$
$\vec{r}$	A munkadarab egy pontját jelöli
r	A munkadarab egy pontjának távolsága a középvonaltól$[m]$
R	A munkadarab felületének távolsága a középvonaltól$[m]$
A	Az amplitúdó
A_{max}	Az amplitúdó maximuma
A_{real}	Az amplitúdónak az a nagysága amit a részecske valójában előre haladt
T_q	Az ω hűtőközeg hőmérséklete $[^\circ C]$
T_c	Számolt hőmérséklet a munkadarab egy pontjában $[^\circ C]$
T_{max}	Maximális hőmérséklet $[^\circ C]$
T_{ini}	Munkadarab kezdeti hőmérséklete $[^\circ C]$

γ	Finomító faktor
ε	Nagyon kicsi, de még pozitív szám
α	Úgynevezett összehúzó-elnyújtó tényező
$\tilde{\alpha}$	Newton-Raphson algoritmus lépés faktora
ω	Hűtőközeg
Π	A vizsgált test
Γ	A vizsgált test felszíne
i, j, k	Futó indexek
K	A számolt lehűlési görbe pontjainak a száma
I	Iteráció száma
$\tilde{w}, \chi$	Súlyértékek
$\vec{x}$	Egy pont helye a keresési térben
$\vec{P}, \vec{G}$	A lokális, valamint globális optimális pozíció koordinátája PSO algotimus esetében
$\vec{p}, \vec{p}_i$	Egy részecske pozíciója, valamint egy részecske pozíciója az i. iterációban
r, u	A $[0 \ldots 1[$ nyílt intervallumon értelmezett egyenletes eloszlású véletlen szám
g	Standard normál eloszlású véletlen szám
N_p, N_g, N_e, N_q	Populációk, Gaussian spark-ok, Explosion spark-ok, Quantum spark-ok száma
$N_T, N_v, N_{\max}$	Tanítóminták száma, Vezérpontok száma, Spark-ok maximális száma
c_1, c_2	Gyorsítási tényezők
$\vec{v}$	Sebességvektor
P	Hőátadási tényező függvény egy vezérlőpontja
$w_{\text{left}}, w_{\text{right}}$	Bal és jobb oldali szélességparaméterek
E	Gradiens módszerrel optimalizálandó függvény
$\tilde{d}$	Deformációs paraméter
$\tilde{p}, \tilde{p}_{\text{left}}, \tilde{p}_{\text{right}}$	Lecsengés paraméterek
$\vec{\pi}_i$	A részecske helyi fókusza

12. Ábrák jegyzéke

13. Táblázatok jegyzéke

14. Algoritmusok jegyzéke

Irodalomjegyzék

Hivatkozott saját releváns publikációk

[48] Zoltán Fried és tsai.: „Parallelized Particle Swarm Optimization to Estimate the Heat Transfer Coefficients of Palm Oil, Canola Oil, Conventional, and Accelerated Petroleum Oil Quenchants”. *Materials Performance and Characterization* 8 (2018), 96–113. old. ISSN: 2379-1365. DOI: 10.1520/MPC20180049.

[90] Zoltán Fried és tsai.: „On the Nature-Inspired Algorithms Applied to Characterize Heat Transfer Coefficients”. *Thermal Processing in Motion*. Spartanburg: ASM, 2018, 47–51. old.

[91] Zoltán Fried és tsai.: „Komplex hőátadási együttható rekonstrukciója bio-inspirált módszer alkalmazásával”. *XXVII. Hőkezelő és anyagtudomány a gépgyártásban országos konferencia és szakkiállítás külföldi részvevőkkel*. 2016, 53–58. old.

[92] I. Felde, Z. Fried és S. Szénási: „Solution of 2-D Inverse Heat Conduction Problem with Graphic Accelerator”. *Materials Performance and Characterization* 6.5 (2017), 882–893. old. ISSN: 2379–1365. DOI: 10.1520/mpc20170008.

[101] Zoltán Fried, Sándor Szénási és Imre Felde: „Prediction of objective function value for heat transfer coefficient function reconstruction by FWA”. *2019 IEEE 13th International Symposium on Applied Computational Intelligence and Informatics (SACI)*. Timisoara, Romania: IEEE, 2019. máj., 305–308. old. ISBN: 978-1-7281-0685-4. DOI: 10.1109/SACI46893.2019.9111623.

Hivatkozott külső forrású irodalom

[1] R. Clausius: „Ueber eine veränderte Form des zweiten Hauptsatzes der mechanischen Wärmetheorie”. *Annalen der Physik* 169.12 (1854. jan.), 481–506. old. DOI: 10.1002/andp.18541691202.

[2] Imre Felde: *A járműipari alkatrészek hőkezeléséhez használthűtőközegek hűtőképességének vizsgálati módszerei*. Kut. jel. Miskolci Egyetem Mechanikai Technológiai Tanszék, 2013.

[3] Shiro Nukiyama: „The maximum and minimum values of the heat Q transmitted from metal to boiling water under atmospheric pressure”. *International Journal of Heat and Mass Transfer* 9.12 (1966), 1419–1433. old. ISSN: 0017-9310. DOI: 10.1016/0017-9310(66)90138-4.

[4] Axel Majorek és tsai.: „Influence of heat transfer on the development of residual stresses in quenched steel cylinders”. *Steel Research* 65.4 (1994. ápr.), 146–151. old. DOI: 10.1002/srin.199400944.

[5] H.M. Tensi és A. Stick: „Martens Hardening of Steel - Prediction of Temperature Distribution and Surface Hardness”. *Materials Science Forum*. Materials Science Forum 102 (1992. jan.), 741–754. old. DOI: 10.4028/www.scientific.net/MSF.102-104.741.

[6] Božidar Liščić, Hans M. Tensi és Wacław Luty, szerk.: *Theory and Technology of Quenching*. Springer Berlin Heidelberg, 1992. DOI: 10.1007/978-3-662-01596-4.

[7] Scott Mackenzie George E. Toten: *Handbook of Aluminium*. Marcel-Dekker, Inc., 2003. ISBN: 0-8247-0494-0.

[8] J. V. Beck, B. Blackwell és C. R. St Clair Jr.: *Inverse Heat Conduction*. Wiley, New York, 1985.

[9] A. N. Tikhonov és V. Y. Arsenin: *Solution of Ill-Posed Problems*. Winston, Washington DC., 1977.

[10] O. M. Alifanov: *Inverse heat transfer problems*. International series in heat and mass transfer. Berlin, New York: Springer-Verlag, 1994. ISBN: 978-3-5405-3679-6.

[11] M. Necati Özisik és Helcio R. B. Orlande: *Inverse Heat Transfer: Fundamentals and Applications*. Routledge, 2018. máj., 352. old. ISBN: 978-0-2037-4978-4. DOI: 10.1201/9780203749784.

[12] K. N. Prabhu és A. A. Ashish: „Inverse modeling of heat transfer with application to solidification and quenching”. *Materials and Manufacturing Processes* 17.4 (2002), 469–481. old.

[13] I. Felde és T. Reti: „Evaluation of hardening performance of cooling media by using inverse heat conduction methods and property prediction”. *Strojniski Vestnik/Journal of Mechanical Engineering* 56.2 (2010. febr.), 77–83. old.

[14] D. Landek, J. Župan és T. Filetin: „A prediction of quenching parameters using inverse analysis”. *Materials Performance and Characterization* 3.2 (2014), 229–241. old.

[15] ISO: *ISO 9950: Industrial Quenching Oils – Determination of Cooling Characteristics Nickel-Alloy Probe Test Method 1995(E)*. ISO, Switzerland, 1995.

[16] J. Clark és R. Tye: „Thermophysical properties reference data for some key engineering alloys”. *High Temperatures – High Pressures* 35–36.1 (2004), 1–14. old. DOI: 10.1068/htjr087.

[17] I. Felde: *Estimation of thermal boundary conditions by gradient based and genetic algorythms*. 729. köt. Trans Tech Publications, 2013, 144–149. old. ISBN: 9783037854914. DOI: 10.4028/www.scientific.net/MSF.729.144.

[18] S. Szénási, I. Felde és I. Kovács: „Solving one-dimensional IHCP with particle swarm optimization using graphics accelerators”. *Proceedings of the 10th IEEE International Symposium on Applied Computational Intelligence and Informatics 2015, Timişoara, Romania* 729 (2015), 365–369. old.

[19] Imre Felde és Sándor Szénási: „Estimation of temporospatial boundary conditions using a particle swarm optimisation technique”. *Int. J. Microstructure and Materials Properties* 11.3/4 (2016), 288–300. old.

[20] Sándor Szénási és Imre Felde: „Configuring Genetic Algorithm to Solve the Inverse Heat Conduction Problem”. *5th International Symposium on Applied Machine Intelligence and Informatics (SAMI2017)*. Herlany: IEEE, 2017, 387–391. old.

[21] Sandor Szenasi és Imre Felde: „Using multiple graphics accelerators to solve the two-dimensional inverse heat conduction problem”. *Computer Methods in Applied Mechanics and Engineering* 336 (2018. márc.). DOI: 10.1016/j.cma.2018.03.024.

[22] Imre Felde: „Liquid quenchant database: determination of heat transfer coefficient during quenching”. *International Journal of Microstructure and Materials Properties* 11.3/4 (2016), 277. old. DOI: 10.1504/IJMMP.2016.079154.

[23] Trevor Hastie, Robert Tibshirani és Jerome Friedman: *The Elements of Statistical Learning*. Springer New York, 2009. DOI: 10.1007/978-0-387-84858-7.

[24] „IEEE Standard for Floating-Point Arithmetic”. *IEEE Std 754-2008* (2008), 1–70. old. DOI: 10.1109/IEEESTD.2008.4610935.

[25] R.E. Bellman: „Dynamic Programming and a new formalism in the calculus of variations”. *Proc. Natl. Acad. Sci.* 40.4 (1954), 231–235. old.

[26] R.E. Bellman: *Dynamic Programming*. Princeton Univ. Press, Princeton, N. J., 1957.

[27] W.R. Hamilton: „On a general method in dynamics”. *Philosophical Transactions of the Royal Society, part II for 1834* (1834), 247–308. old.

[28] W.R. Hamilton: „Second essay on a general method in dynamic”. *Philosophical Transactions of the Royal Society, part I for 1835* (1835), 95–144. old.

[29] S. Boyd és tsai.: *Linear Matrix Inequalities in Systems and Control Theory*. SIAM books, Philadelphia, 1994.

[30] HS Carlsaw és JC Jaeger: „Conduction of heat in solids”. *Clarendon, Oxford* 2nd (1959).

[31] David Langford: „New analytic solutions of the one-dimensional heat equation for temperature and heat flow rate both prescribed at the same fixed boundary (with applications to the phase change problem)”. *Quarterly of Applied Mathematics* 24.4 (1967. jan.), 315–322. old. DOI: https://doi.org/10.1090/qam/211094.

[32] John Crepeau: „Josef Stefan and His Contributions to Heat Transfer”. *Heat Transfer: Volume 3*. ASMEDC, 2008. jan. DOI: 10.1115/HT2008-56073.

[33] J. Stefan: „Ueber die Theorie der Eisbildung, insbesondere über die Eisbildung im Polarmeere (On the theory of ice formation, especially on ice formation in polar seas)”. *Annalen der Physik* 278.2 (1891), 269–286. old. DOI: 10.1002/andp.18912780206.

[34] O. R. Burggraf: „An Exact Solution of the Inverse Problem in Heat Conduction Theory and Applications”. *Journal of Heat Transfer* 86.3 (1964. aug.), 373–380. old. DOI: 10.1115/1.3688700.

[35] A. G. Tyomkin: „Determination of time dependent external heat actions by methods of heat conduction problem (in Russian), Izv. Vyssh. Uchebn. Zaved., Ser. Energetika”. No. 5 (1961), 60–71. old.

[36] Jr. Stolz G.: „Numerical Solutions to an Inverse Problem of Heat Conduction for Simple Shapes”. *Journal of Heat Transfer* 82.1 (1960. febr.), 20–25. old. ISSN: 0022-1481. DOI: 10.1115/1.3679871.

[37] A. N. Tikhonov: „Solution of incorrectly formulated problems and the regularization method”. *Soviet Math. Dokl.* 4 (1963), 1035–1038. old.

[38] A.N. Tikhonov és V.B. Glasko: „Use of the regularization method in non-linear problems”. *USSR Computational Mathematics and Mathematical Physics* 5.3 (1965. jan.), 93–107. old. DOI: 10.1016/0041-5553(65)90150-3.

[39] A. N. Tikhonov és tsai.: *Numerical Methods for the Solution of Ill-Posed Problems*. Springer Netherlands, 1995. DOI: 10.1007/978-94-015-8480-7.

[40] E. A. Artyukhin & A. V. Nenarokomov O.M. Alifanov: „Spline approximation of the solution of the inverse heat-conduction problem,taking account of the smoothness of the desiredfunction”. *TVT* 25.4 (1987), 693–699. old.

[41] L. A. Kozdoba V. L. Chumakov: *Solution of nonlinear problems of nonstationary heat conduction by the perturbation method*. Szerk. TVT. Naukova Dumka, 1976.

[42] M. Matsevity: *Electric simulation of nonlinear problems in engineering thermodynamics*. Naukova Dumka, 1977.

[43] B. Heinrich és tsai.: „Inverse Heat Conduction. Ill-Posed Problems". *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik* 67.3 (1987), 212–213. old. DOI: 10.1002/zamm.19870670331.

[44] V. V. Glasko A.A Tikhonov: „Methods of determining the surface temperature of a body". *USSR Computational Mathematics and Mathematical Physics* 7.4 (1967. jan.), 267–273. old. DOI: 10.1016/0041-5553(67)90161-9.

[45] O. M. Alifanov és V. V. Mikhailov: „Solution of the overdetermined inverse problem of thermal conductivity involving inaccurate data". *Teplofizika Vysokikh Temperatur* 23 (1985. júl.), 120–125. old.

[46] Swati Verma és C. Balaji: „Multi-parameter estimation in combined conduction-radiation from a plane parallel participating medium using genetic algorithms". *International Journal of Heat and Mass Transfer* 50.9-10 (2007), 1706–1714. old. ISSN: 0017-9310. DOI: 10.1016/j.ijheatmasstransfer.2006.10.045.

[47] Ki Wan Kim és Seung Wook Baek: „Inverse surface radiation analysis in an axisymmetric cylindrical enclosure using a hybrid genetic algorithm". *Numerical Heat Transfer, Part A: Applications* 46.4 (2004. aug.), 367–381. old. DOI: 10.1080/10407780490478533.

[49] Marco Dorigo és Mauro Birattari: „Ant Colony Optimization". *Encyclopedia of Machine Learning*. Szerk. Claude Sammut és Geoffrey I. Webb. Boston, MA: Springer US, 2010, 36–39. old. ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_22.

[50] J. Kennedy és R. Eberhart: „Particle swarm optimization". *Neural Networks, 1995. Proceedings., IEEE International Conference on* 4 (1995), 1942–1948. old. ISSN: 1935-3812. DOI: 10.1109/ICNN.1995.488968.

[51] Dušan Teodorović: „Bee Colony Optimization (BCO)". *Innovations in Swarm Intelligence*. Szerk. Chee Peng Lim, Lakhmi C. Jain és Satchidananda Dehuri. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, 39–60. old. ISBN: 978-3-642-04225-6. DOI: 10.1007/978-3-642-04225-6_3.

[52] Ying Tan és Yuanchun Zhu: „Fireworks Algorithm for Optimization". *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2010, 355–364. old. DOI: 10.1007/978-3-642-13495-1_44.

[53] Ying Tan: „Fireworks Algorithm (FWA)”. *Fireworks Algorithm*. Springer Berlin Heidelberg, 2015, 17–35. old. ISBN: 978-3-662-46353-6. DOI: 10.1007/978-3-662-46353-6_2.

[54] J. Sun, C.-H Lai és X.-J Wu: *Particle swarm optimisation: Classical and quantum perspectives*. CRC Press, 2016. ápr., 1–393. old. DOI: 10.1201/b11579.

[55] Jeff Bezanson és tsai.: *Julia: A Fresh Approach to Numerical Computing*. 2015.

[56] Manish Arora: „The Architecture and Evolution of CPU-GPU Systems for General Purpose Computing”. 2012.

[57] András Molnár és Dániel Stojcsics: „GPGPU accelerated large scale 3D object reconstruction application for fixed and rotary wing unmanned aerial vehicles”. *International Journal of Systems, Applications, Engineering & Development* 10 (2016), 189–194. old.

[58] Attila Reményi és tsai.: „Parallel biomedical image processing with GPGPUs in cancer research”. *3rd IEEE International Symposium on Logistics and Industrial Informatics (LINDI 2011)*. Budapest: IEEE, 2011, 245–248. old. ISBN: 978-1-4577-1840-3.

[59] Sándor Szénási, Zoltán Vámossy és Miklós Kozlovszky: „GPGPU-based data parallel region growing algorithm for cell nuclei detection”. *12th International Symposium on Computational Intelligence and Informatics (CINTI2011)*. Budapest: IEEE, 2011, 493–499. old. ISBN: 978-1-4577-0044-6. DOI: 10.1109/CINTI.2011.6108556.

[60] Zoltán Kerekes és tsai.: „Colon cancer diagnosis on digital tissue images”. *IEEE 9th International Conference on Computational Cybernetics (ICCC2013)*. Tihany: IEEE, 2013, 159–163. old. ISBN: 978-1-4799-0060-2. DOI: 10.1109/ICCCyb.2013.6617580.

[61] NVIDIA: *CUDA C Programming Guide*. 2014.

[62] Jason Sanders és Edward Kandrot: *CUDA by Example*. Addison Wesley, 2010, 279. old. ISBN: 9780131387683. DOI: 10.5555/1891996.

[63] John Stone, David Gohara és Guochun Shi: „OpenCL: A Parallel Programming Standard for Heterogeneous Computing Systems”. *Computing in science & engineering* 12 (2010. máj.), 66–72. old. DOI: 10.1109/MCSE.2010.69.

[64] Shane Cook: *CUDA Programming*. Morgan Kaufmann, 2012. ISBN: 9780124159334. DOI: 10.1016/B978-0-12-415933-4.00010-7.

[65] S. Szénási és I. Felde: „GPU-based Heat Transfer Model”. *International Multidisciplinary Scientific GeoConference Surveying Geology and Mining Ecology Management (SGEM2017)*. Albena, 2017, 343–350. old. DOI: 10.5593/sgem2017/21/S07.044.

[66] Sándor Szénási és Imre Felde: „Heat Transfer Simulation using GPUs”. *20th IEEE Jubilee International Conference on Intelligent Engineering Systems (INES2016)*. Budapest: IEEE, 2016, 263–267. old.

[67] Randall LeVeque: *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems (Classics in Applied Mathematics Classics in Applied Mathemat)*. USA: Society for Industrial és Applied Mathematics, 2007. ISBN: 0898716292.

[68] Hans Petter Langtangen és Svein Linge: *Finite Difference Computing with PDEs*. Springer International Publishing, 2017. DOI: 10.1007/978-3-319-55456-3.

[69] Olusegun Adeyemi Olaiju, Yeak Hoe és Ezekiel Ogunbode: „Finite-Difference Approximations to the Heat Equation via C”. *Journal of Applied Sciences & Environmental Sustainability* 3 (2017. júl.), 188–200. old.

[70] Siu Chin: „A theory of explicit finite-difference schemes”. (2013. szept.).

[71] J. Crank és P. Nicolson: „A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type”. *Advances in Computational Mathematics* 6.1 (1996. dec.), 207–226. old. DOI: 10.1007/bf02127704.

[72] Anthony Ralston: „Runge-Kutta methods with minimum error bounds”. *Mathematics of Computation* 16.80 (1962), 431–431. old. DOI: 10.1090/s0025-5718-1962-0150954-0.

[73] Joe D. Hoffman, Joe D. Hoffman és Steven Frankel: *Numerical Methods for Engineers and Scientists*. CRC Press, 2018. okt. DOI: 10.1201/9781315274508.

[74] C. E. Shannon: „A Mathematical Theory of Communication”. *Bell System Technical Journal* 27.3 (1948. júl.), 379–423. old. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[75] C.E. Shannon: „Communication in the Presence of Noise”. *Proceedings of the IRE* 37.1 (1949. jan.), 10–21. old. DOI: 10.1109/jrproc.1949.232969.

[76] Kevin P. Murphy: *Probabilistic Machine Learning: An introduction*. MIT Press, 2022.

[77] M. Raudenský, J. Horský és J. Krejsa: „Usage of neural network for coupled parameter and function specification inverse heat conduction problem”. *International Communications in Heat and Mass Transfer* 22.5 (1995. szept.), 661–670. old. DOI: 10.1016/0735-1933(95)00052-z.

[78] J. Krejsa és tsai.: „Assessment of strategies and potential for neural networks in the inverse heat conduction problem”. *Inverse Problems in Engineering* 7.3 (1999. jún.), 197–213. old. DOI: 10.1080/174159799088027694.

[79] S.S. Sablani és tsai.: „Non-iterative estimation of heat transfer coefficients using artificial neural network models”. *International Journal of Heat and Mass Transfer* 48.3-4 (2005. jan.), 665–679. old. DOI: 10.1016/j.ijheatmasstransfer.2004.09.005.

[80] S. Deng és Y. Hwang: „Applying neural networks to the solution of forward and inverse heat conduction problems”. *International Journal of Heat and Mass Transfer* 49.25-26 (2006. dec.), 4732–4750. old. DOI: 10.1016/j.ijheatmasstransfer.2006.06.009.

[81] Lexin Zhang és tsai.: „Time-Series Neural Network: A High-Accuracy Time-Series Forecasting Method Based on Kernel Filter and Time Attention”. *Information* 14.9 (2023. szept.), 500. old. ISSN: 2078-2489. DOI: 10.3390/info14090500.

[82] Věra Kůrková és Marcello Sanguineti: „Can Two Hidden Layers Make a Difference?”: *Adaptive and Natural Computing Algorithms*. Springer Berlin Heidelberg, 2013, 30–39. old. ISBN: 9783642372131. DOI: 10.1007/978-3-642-37213-1_4.

[83] G. Brightwell, C. Kenyon és H. Paugam-Moisy: „Multilayer neural networks: one or two hidden layers?”: *Proceedings of the 9th International Conference on Neural Information Processing Systems*. NIPS’96. Denver, Colorado: MIT Press, 1996, 148–154. old.

[84] Kamal Sikka, Rahul Lall és Tuhin Sinha: „Artificial Neural Networks for Package Thermal Analysis”. *2021 20th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (iTherm)*. 32. köt. IEEE, 2021. jún., 103–113. old. DOI: 10.1109/itherm51669.2021.9503167.

[85] Ahmad Alwosheel, Sander van Cranenburgh és Caspar G. Chorus: „Is your dataset big enough? Sample size requirements when using artificial neural networks for discrete choice analysis”. *Journal of Choice Modelling* 28 (2018. szept.), 167–182. old. ISSN: 1755-5345. DOI: 10.1016/j.jocm.2018.07.002.

[86] Akos Szabo-Gali és tsai.: „Approximation of Heat Transfer Coefficients by using AI techniques”. *Japan Society for Heat Treatment*. IFHTSE, 2023.

[87] Sandor Szenasi és tsai.: „Estimating the Heat Transfer Coefficient Using Universal Function Approximator Neural Network”. *2018 IEEE 12th International Symposium on Applied Computational Intelligence and Informatics (SACI)*. 13. köt. IEEE, 2018. máj., 401–000404. old. DOI: 10.1109/saci.2018.8440928.

[88] S. Vakili és Mohamed S. Gadala: „Effectiveness and Efficiency of Particle Swarm Optimization Technique in Inverse Heat Conduction Analysis”. *NUMERICAL HEAT TRANSFER PART B-FUNDAMENTALS* 56.2 (2009), 119–141. old. ISSN: 1040-7790. DOI: 10.1080/10407790903116469.

[89] Obed Cortés-Aburto, Rafael Rojas-Rodríguez és Rita Marina Aceves-Pérez: „Inverse Heat Transfer Using Particle Swarm Optimization Methods for Heat Source Estimation”. *Experimental and Theoretical Advances in Fluid Dynamics*. Szerk. Jaime Klapp és tsai. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, 361–369. old. ISBN: 978-3-642-17958-7. DOI: 10.1007/978-3-642-17958-7_31.

[93] Radha Thangaraj és tsai.: „Particle swarm optimization: Hybridization perspectives and experimental illustrations”. *Applied Mathematics and Computation* 217.12 (2011. febr.), 5208–5226. old. DOI: 10.1016/j.amc.2010.12.053.

[94] Yuhui Shi és Russell C. Eberhart: „Parameter selection in particle swarm optimization”. *Evolutionary Programming VII*. Szerk. V. W. Porto és tsai. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, 591–600. old. ISBN: 978-3-540-68515-9. DOI: 10.1007/BFb0040810.

[95] M. Clerc: „The swarm and the queen: towards a deterministic and adaptive particle swarm optimization”. *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*. IEEE, 1999, 1951–1957. old. DOI: 10.1109/cec.1999.785513.

[96] M. Clerc és J. Kennedy: „The particle swarm - explosion, stability, and convergence in a multidimensional complex space”. *IEEE Transactions on Evolutionary Computation* 6.1 (2002), 58–73. old. DOI: 10.1109/4235.985692.

[97] Fung-Bao Liu: „Particle Swarm Optimization-based algorithms for solving inverse heat conduction problems of estimating surface heat flux”. *International Journal of Heat and Mass Transfer* 55.7-8 (2012. márc.), 2062–2068. old. DOI: 10.1016/j.ijheatmasstransfer.2011.12.007.

[98] Jun Sun, Wenbo Xu és Bin Feng: „A global search strategy of quantum-behaved particle swarm optimization”. *IEEE Conference on Cybernetics and Intelligent Systems, 2004.* IEEE, 2004. DOI: 10.1109/iccis.2004.1460396.

[99] Jun Sun, Bin Feng és Wenbo Xu: „Particle swarm optimization with particles having quantum behavior”. *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No.04TH8753)*. IEEE, 2004. DOI: 10.1109/cec.2004.1330875.

[100] Jun Sun, Wenbo Xu és Bin Feng: „Adaptive Parameter Control for Quantum-behaved Particle Swarm Optimization on Individual Level”. *2005 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 2006. DOI: 10.1109/icsmc.2005.1571614.

[102] Ying Tan: *Enhanced Fireworks Algorithm*. 2015. DOI: 10.1007/978-3-662-46353-6_6.

[103] Ke Ding és Ying Tan: „Attract-Repulse Fireworks Algorithm and its CUDA Implementation Using Dynamic Parallelism”. *International Journal of Swarm Intelligence Research* 6.2 (2015. ápr.), 1–31. old. DOI: 10.4018/ijsir.2015040101.

[104] Ying Tan: *Adaptive Fireworks Algorithm*. 2015. DOI: 10.1007/978-3-662-46353-6_8.

[105] Ying Tan: „Cooperative Fireworks Algorithm”. *Fireworks Algorithm: A Novel Swarm Intelligence Optimization Method*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, 133–149. old. ISBN: 978-3-662-46353-6. DOI: 10.1007/978-3-662-46353-6_9.

[106] Chao Yu, Junzhi Li és Ying Tan: „Improve enhanced fireworks algorithm with differential mutation”. San Diego, CA. San Diego, CA: IEEE, 2014. okt., 264–269. old. ISBN: 978-1-4799-3840-7. DOI: 10.1109/SMC.2014.6973918.

[107] Vijay Kumar, Jitender Kumar Chhabra és Dinesh Kumar: „Optimal Choice of Parameters for Fireworks Algorithm”. *Procedia Computer Science* 70 (2015), 334–340. old. DOI: 10.1016/j.procs.2015.10.027.

[108] Xi-Guang Li és tsai.: „Adaptive Mutation Dynamic Search Fireworks Algorithm”. *Algorithms* 10.2 (2017. ápr.), 48. old. DOI: 10.3390/a10020048.

[109] Shaoqiu Zheng és tsai.: „A Cooperative Framework for Fireworks Algorithm”. (2015). DOI: 10.48550/ARXIV.1505.00075.

[110] Jun Sun és tsai.: „Quantum-behaved particle swarm optimization with Gaussian distributed local attractor point”. *Applied Mathematics and Computation* 218.7 (2011), 3763–3775. old. ISSN: 0096-3003. DOI: https://doi.org/10.1016/j.amc.2011.09.021.

[111] Maolong Xi és tsai.: „Improved quantum-behaved particle swarm optimization with local search strategy”. *Journal of Algorithms & Computational Technology* 11 (2016. jún.). DOI: 10.1177/1748301816654020.

[112] Yuming Peng, Yi Xiang és Yubin Zhong: „Quantum-behaved particle swarm optimization algorithm with Lévy mutated global best position”. *2013 Fourth International Conference on Intelligent Control and Information Processing (ICICIP)*. 2013, 529–534. old. DOI: 10.1109/ICICIP.2013.6568132.

[113] J. Y. Shin, M. S. Kim és S. T. Ro: „Correlation of Evaporative Heat Transfer Coefficients for Refrigerant Mixtures”. *Proc. of the International Refrigeration and Air Conditioning Conference 1996* Paper 316. (1996), 151–156. old.

[114] A. Cavallini és tsai.: „Heat Transfer Coefficients Of HFC Refrigerants During Condensation At High Temperature Inside An Enhanced Tube”. *Proc. of the International Refrigeration and Air Conditioning Conference 2002* Paper 563. (2002), 10. old.

[115] R. Mastrullo és tsai.: „Comparison of R744 and R134a heat transfer coefficients during flow boiling in a horizontal circular smooth tube”. *Proc. of the International Conference on Renewable Energies and Power Quality (ICREPQ'09), 15th to 17th April, 2009, Valencia, Spain* 1.7 (2009), 577–581. old.

[116] J. Dombi és T. J.ónás: „Approximations to the normal probability distribution function using operators of continuous-valued logic”. *Acta Cybernetica* 23.3 (2018), 829–852. old.

[117] J. Dombi, T. J.ónás és Z. E. T.óth: „The epsilon probability distribution and its application in reliability theory”. *Acta Polytechnica Hungarica* 15.1 (2018), 197–216. old.

[118] J. Dombi és tsai.: „The omega probability distribution and its applications in reliability theory”. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 35.2 (2019), 600–626. old.

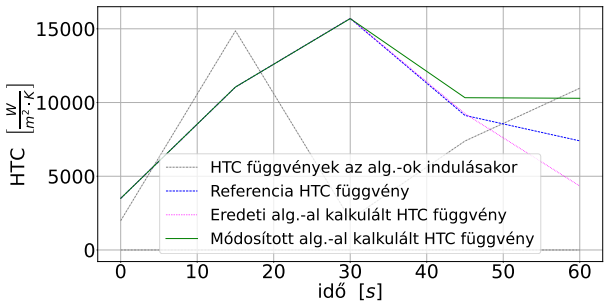
[119] J. Dombi és T. J.ónás: „Kappa regression: an alternative to logistic regression”. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* accepted paper, in press (2019).

[120] J.L. Lagrange, J.P.M. Binet és J.G. Garnier: *Mécanique analytique (Eds. J.P.M. Binet and J.G. Garnier)*. Ve Courcier, Paris, 1811.

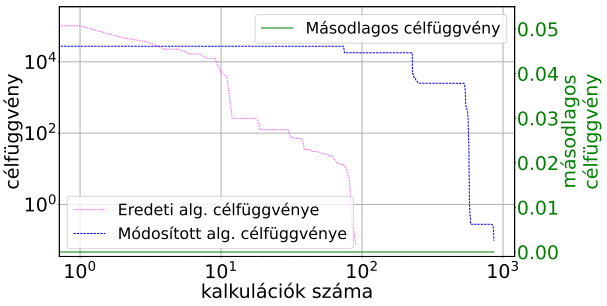
[121] Tjalling J. Ypma: „Historical development of the Newton-Raphson method”. *SIAM Review* 37.4 (1995), 531–551. old.

Mellékletek

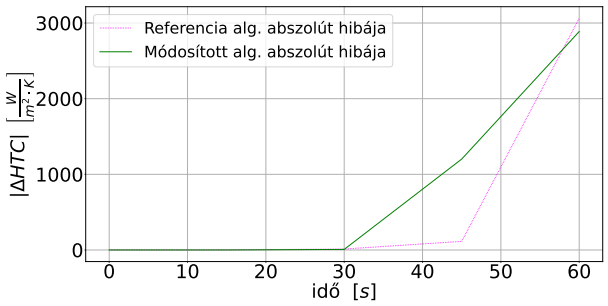
I. Melléklet - PSO algoritmusok futási eredményei



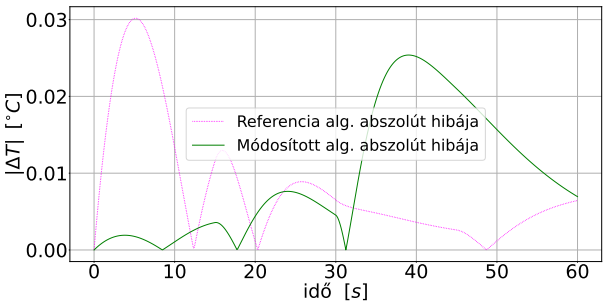
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

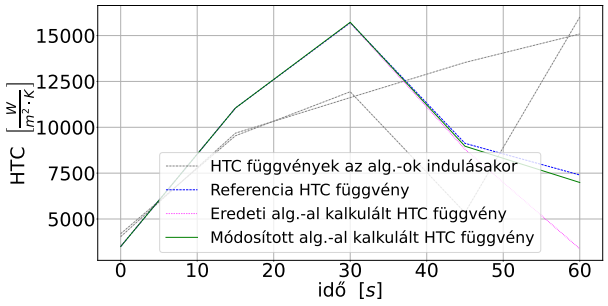


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

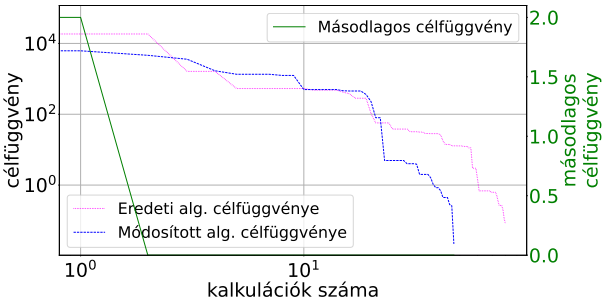


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

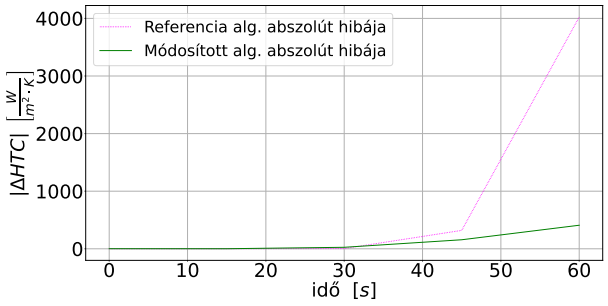
M1. ábra. A PSO algoritmus futtatásának eredménye 5 dimenziós keresési térben



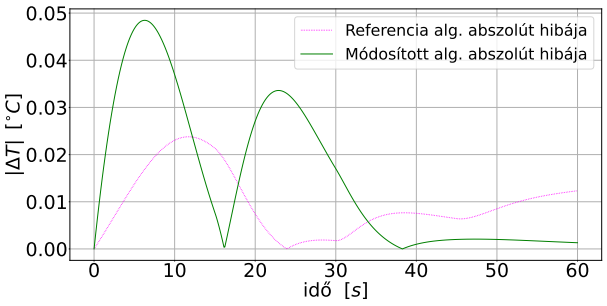
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

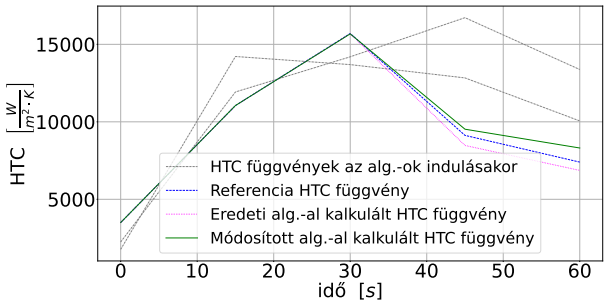


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

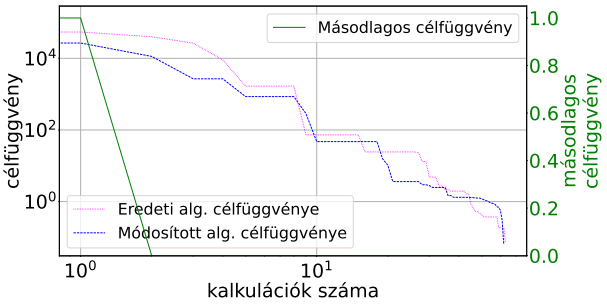


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

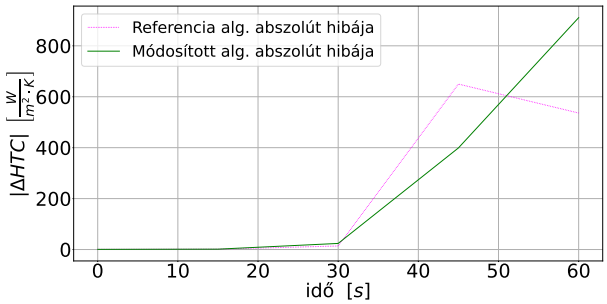
M2. ábra. A PSOCo algoritmus futtatásának eredménye 5 dimenziós keresési térben



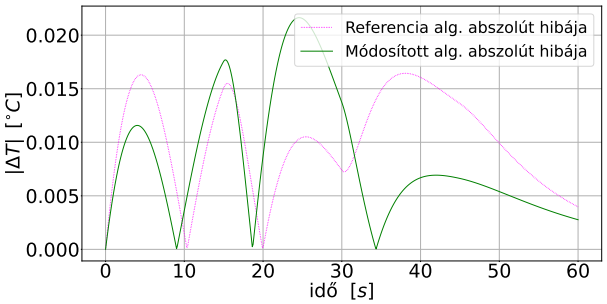
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

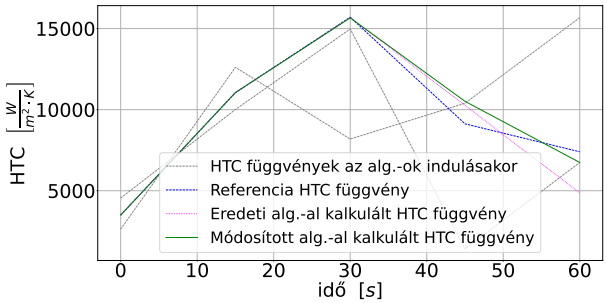


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

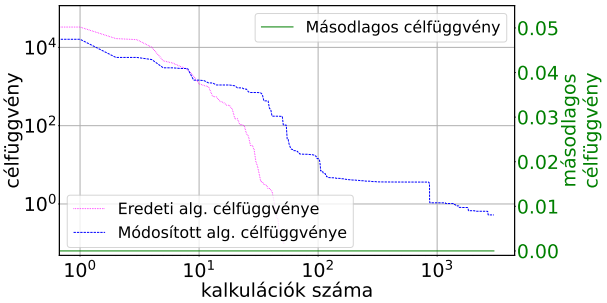


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

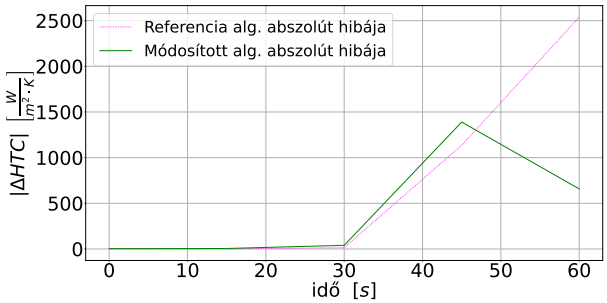
M3. ábra. A PSOIn algoritmus futtatásának eredménye 5 dimenziós keresési térben



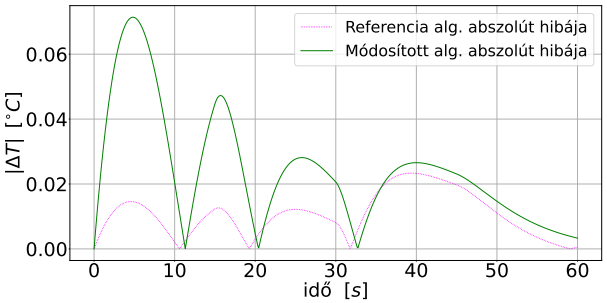
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

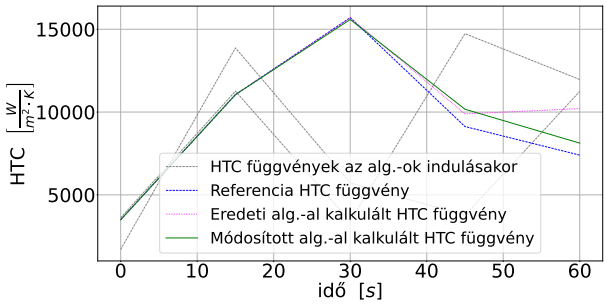


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

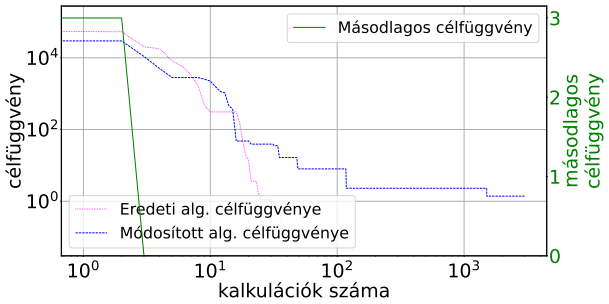


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

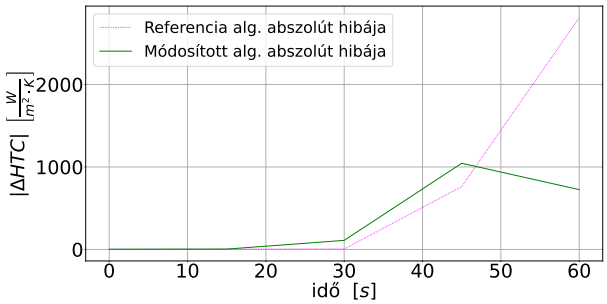
M4. ábra. A QPSOT1 algoritmus futtatásának eredménye 5 dimenziós keresési térben



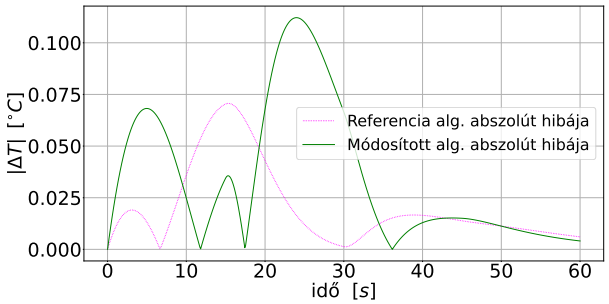
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

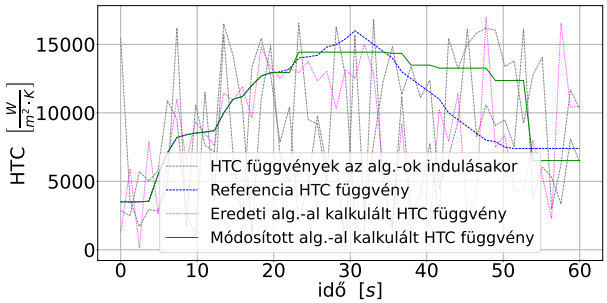


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

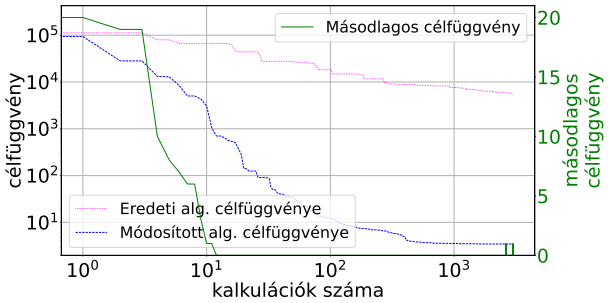


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

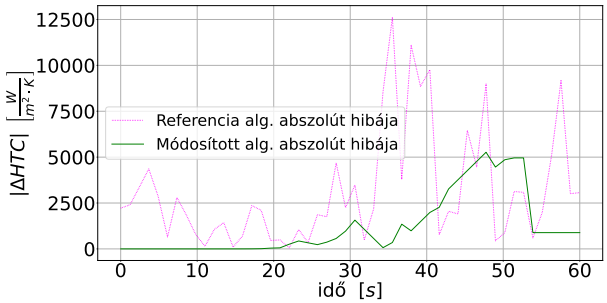
M5. ábra. A QPSOT2 algoritmus futtatásának eredménye 5 dimenziós keresési térben



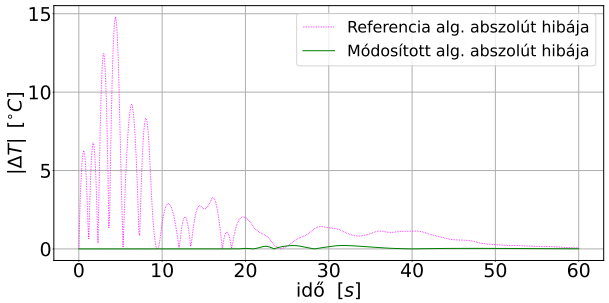
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

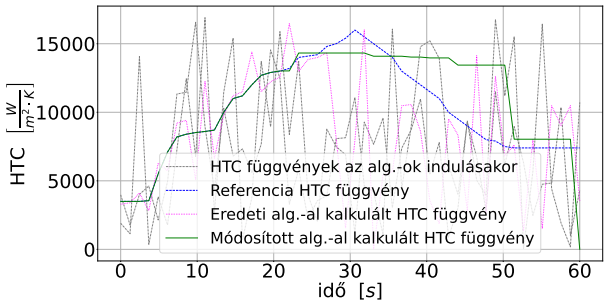


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

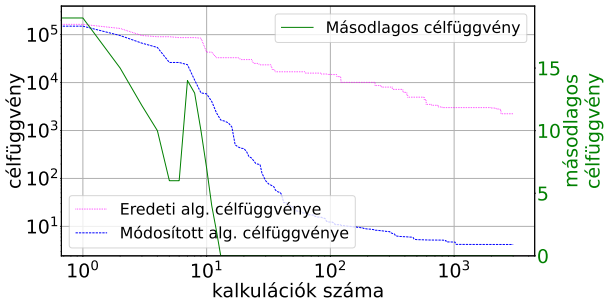


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

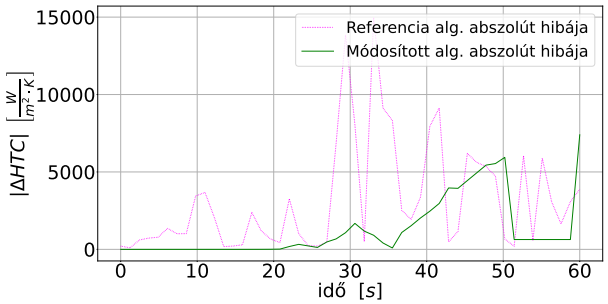
M6. ábra. A PSO algoritmus futtatásának eredménye 50 dimenziós keresési térben



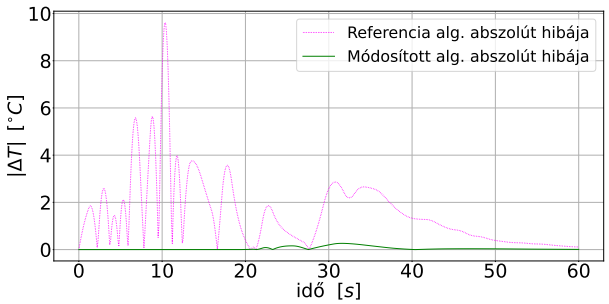
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

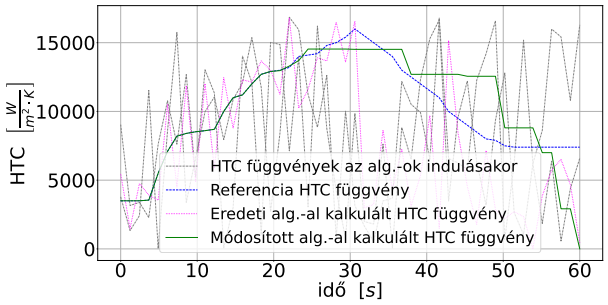


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

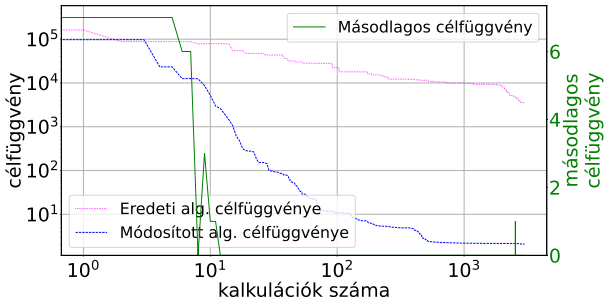


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

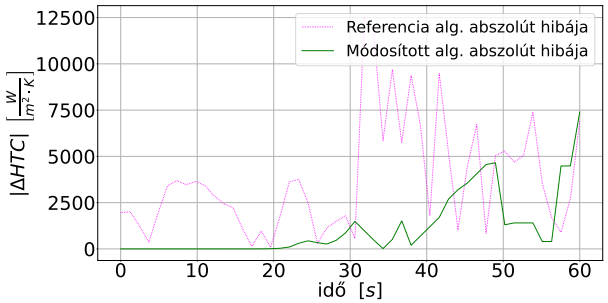
M7. ábra. A PSOCo algoritmus futtatásának eredménye 50 dimenziós keresési térben



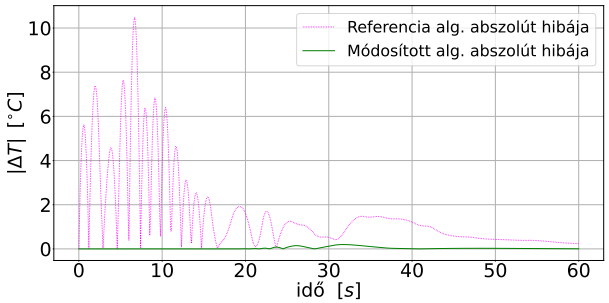
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

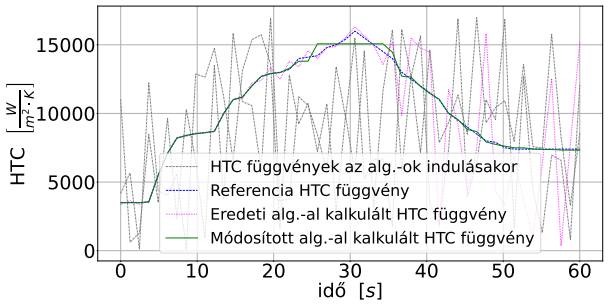


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

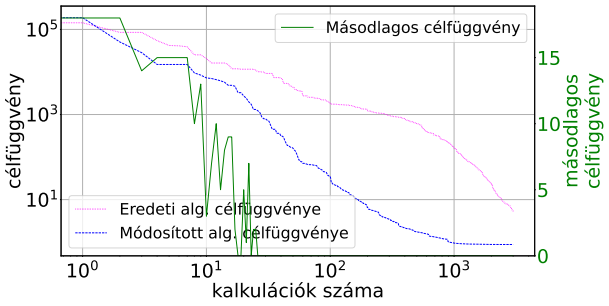


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

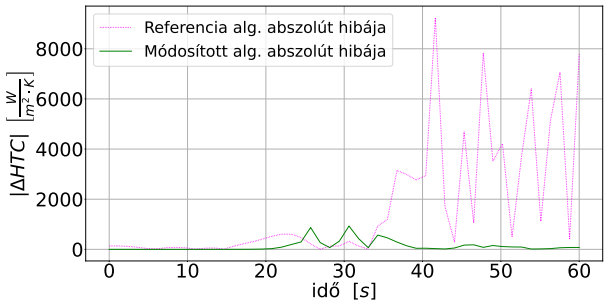
M8. ábra. A PSOIn algoritmus futtatásának eredménye 50 dimenziós keresési térben



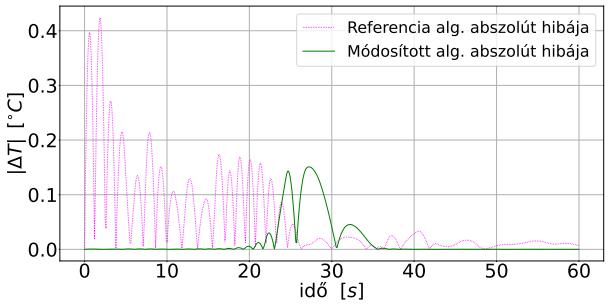
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

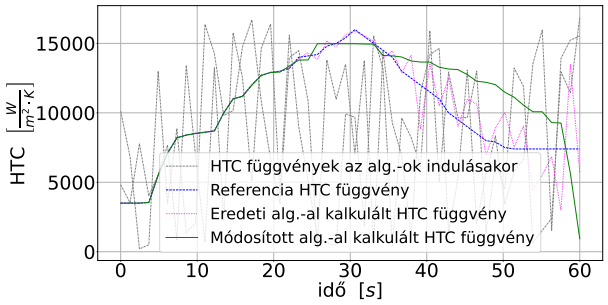


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

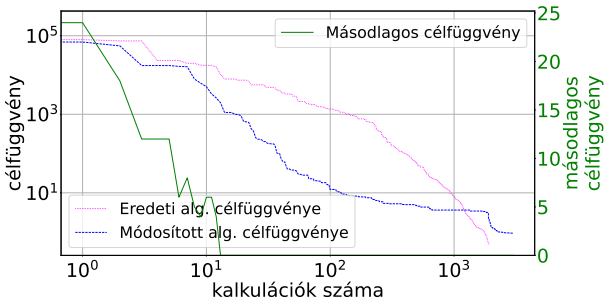


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

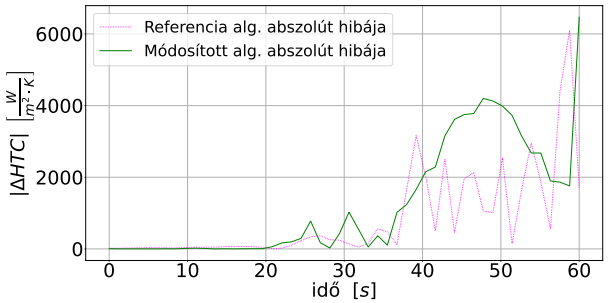
M9. ábra. A QPSOT1 algoritmus futtatásának eredménye 50 dimenziós keresési térben



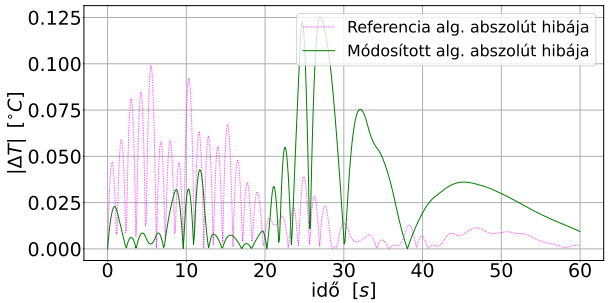
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

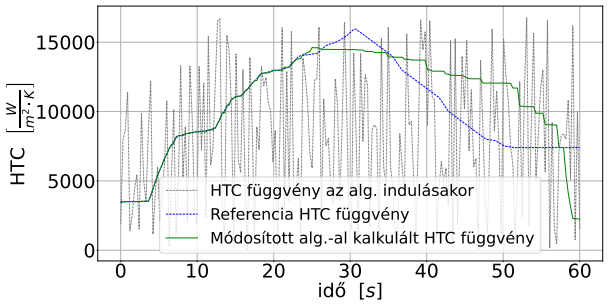


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

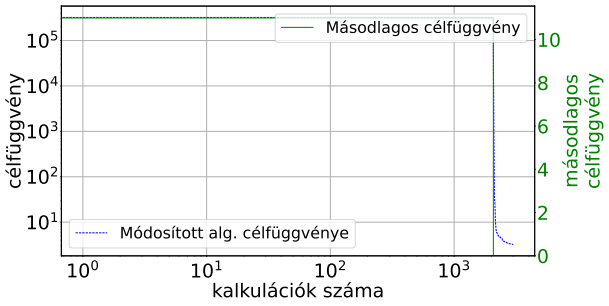


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

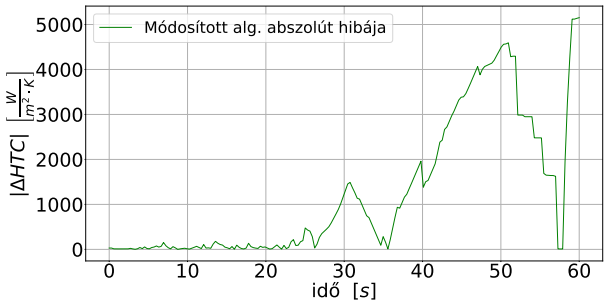
M10. ábra. A QPSOT2 algoritmus futtatásának eredménye 50 dimenziós keresési térben



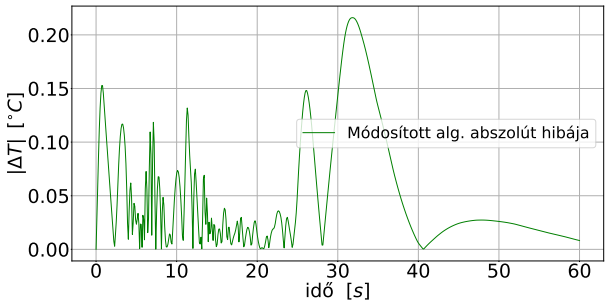
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

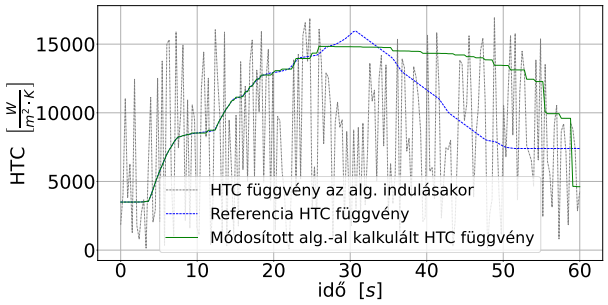


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

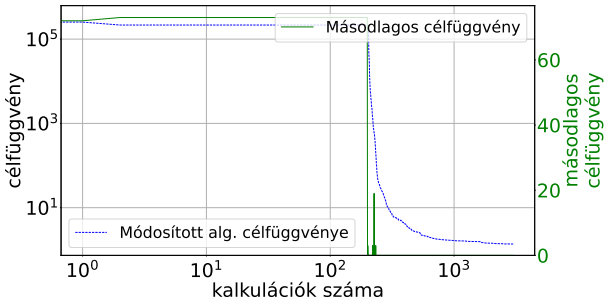


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

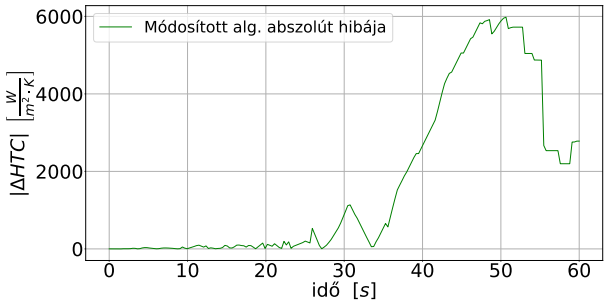
M11. ábra. A PSO algoritmus futtatásának eredménye 200 dimenziós keresési térben



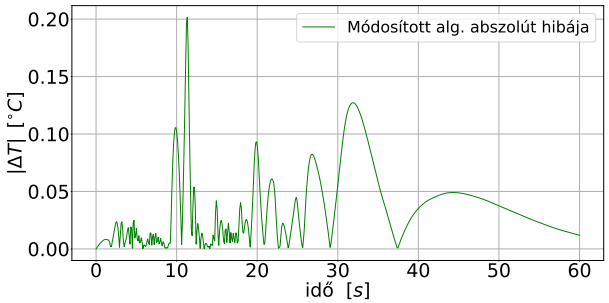
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

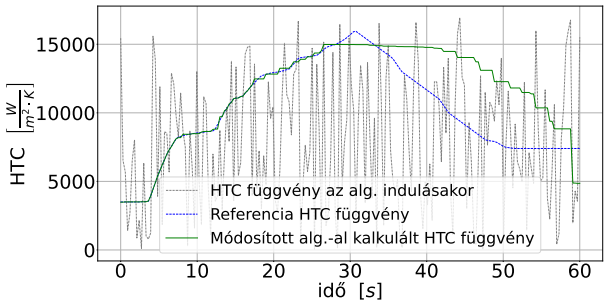


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

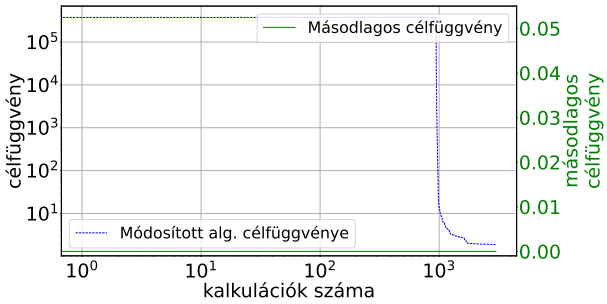


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

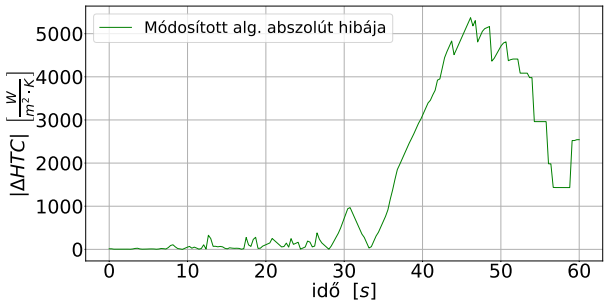
M12. ábra. A PSOCo algoritmus futtatásának eredménye 200 dimenziós keresési térben



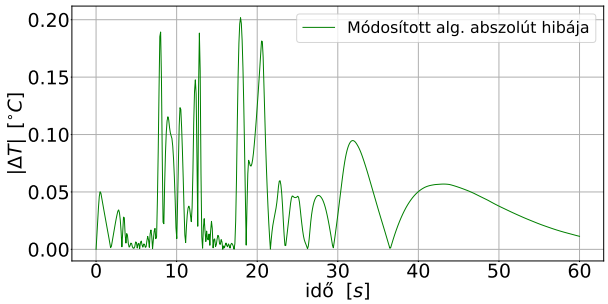
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

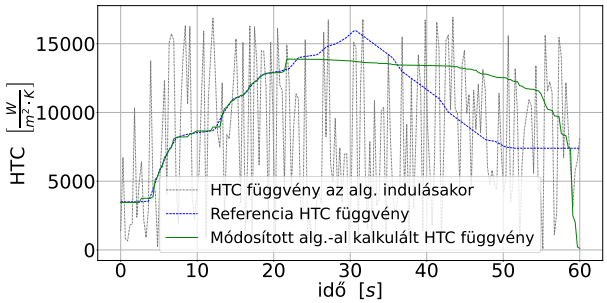


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

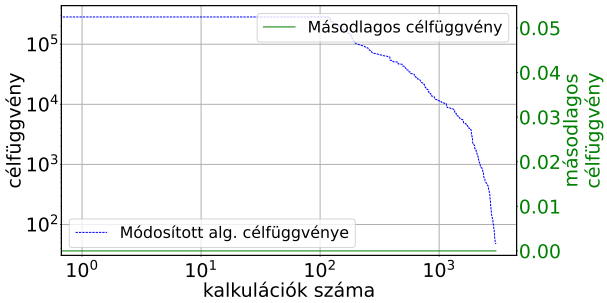


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

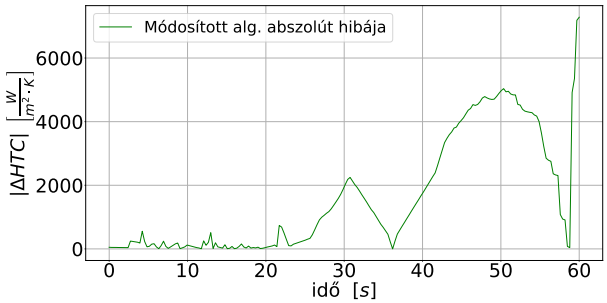
M13. ábra. A PSOIn algoritmus futtatásának eredménye 200 dimenziós keresési térben



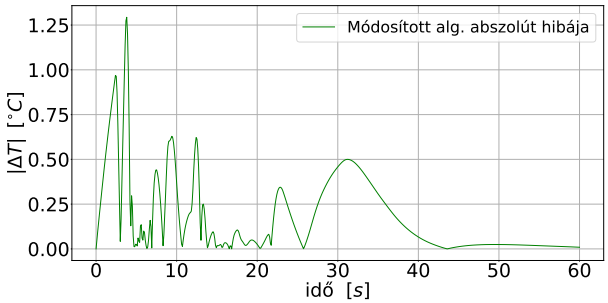
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

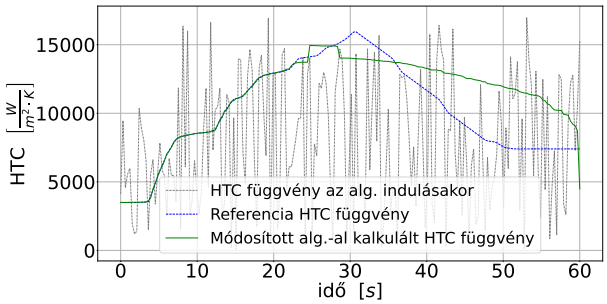


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

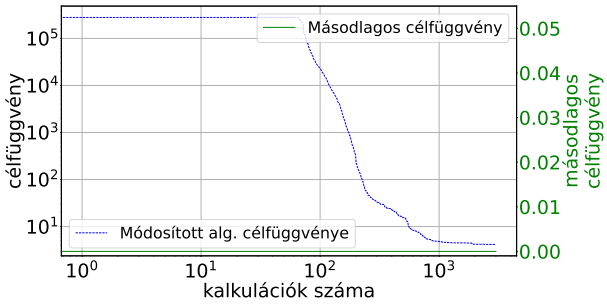


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

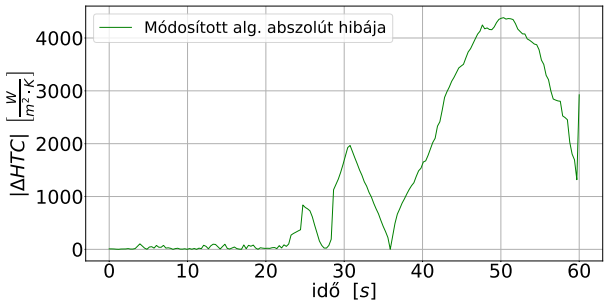
M14. ábra. A QPSOT1 algoritmus futtatásának eredménye 200 dimenziós keresési térben



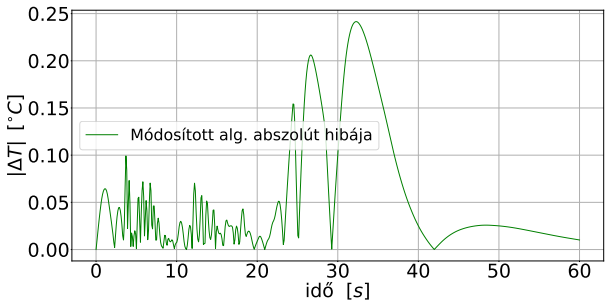
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása



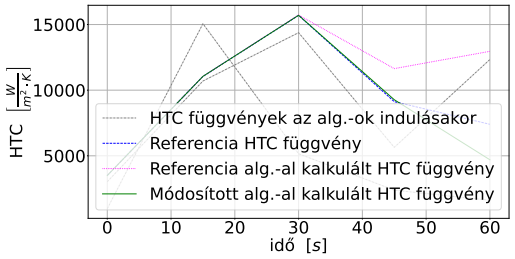
(c) Kalkulált HTC függvényének eltérése a referencia görbétől



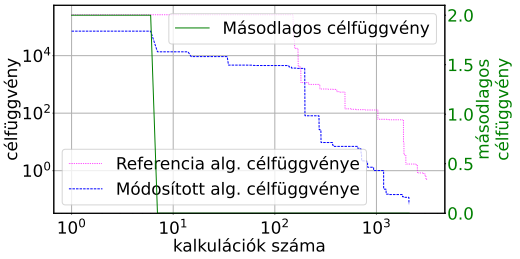
(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

M15. ábra. A QPSOT2 algoritmus futtatásának eredménye 200 dimenziós keresési térben

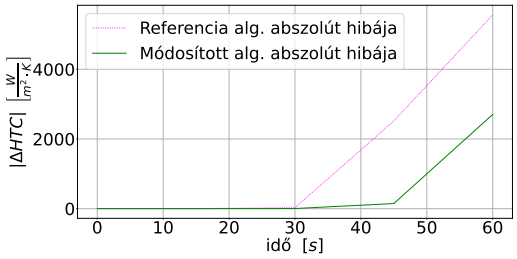
II. Melléklet - FWA algoritmusok futási eredményei



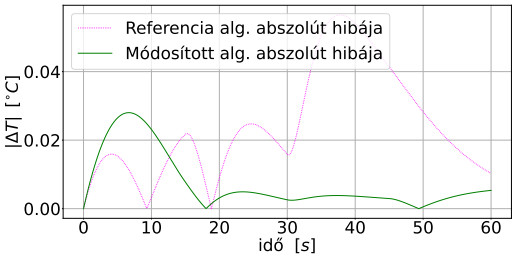
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

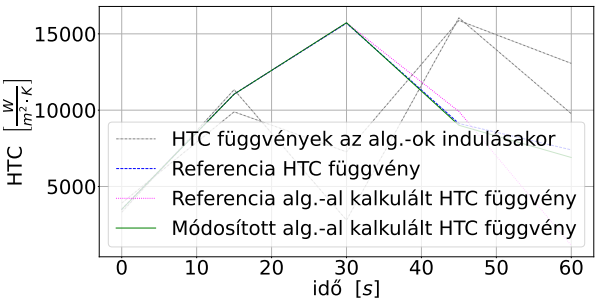


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

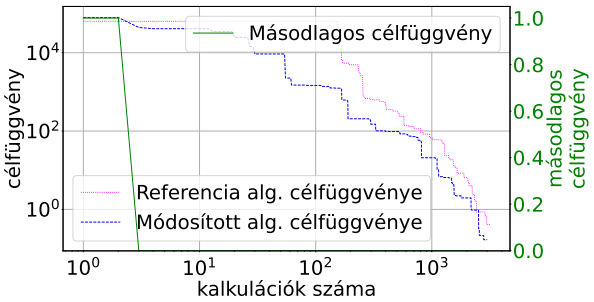


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

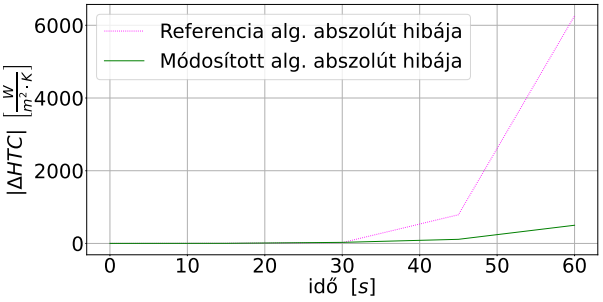
M16. ábra. A FWA algoritmus futtatásának eredménye 5 dimenziós keresési térben



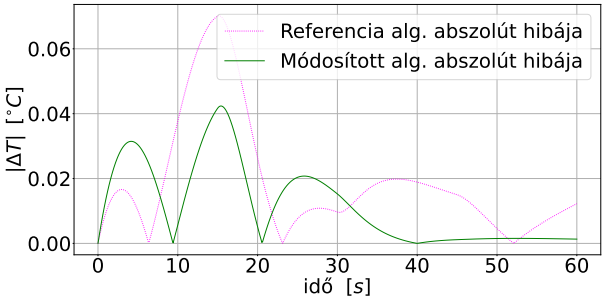
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

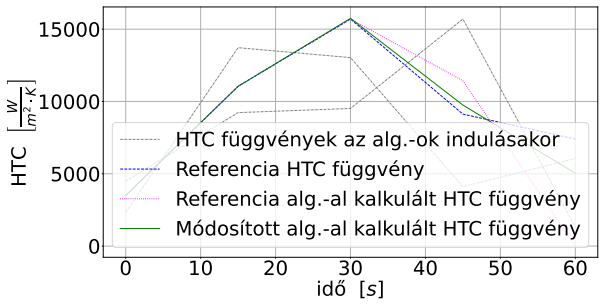


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

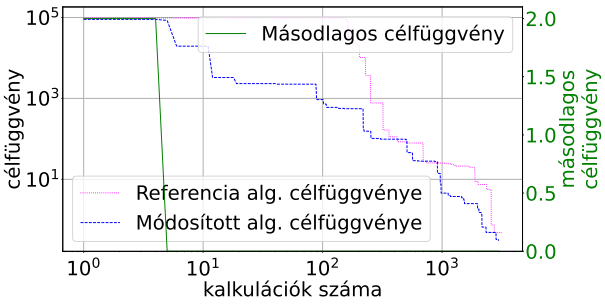


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

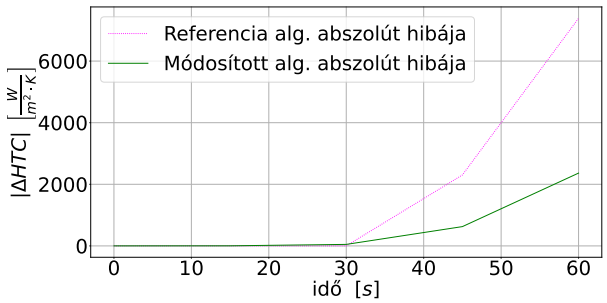
M17. ábra. Az AFWA algoritmus futtatásának eredménye 5 dimenziós keresési térben



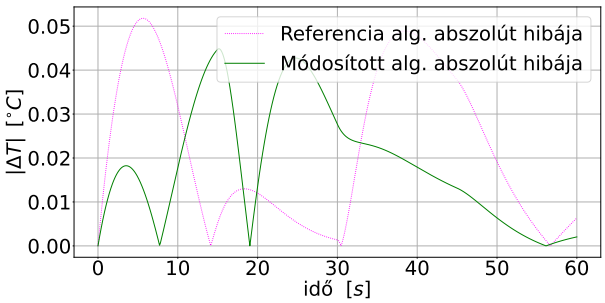
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

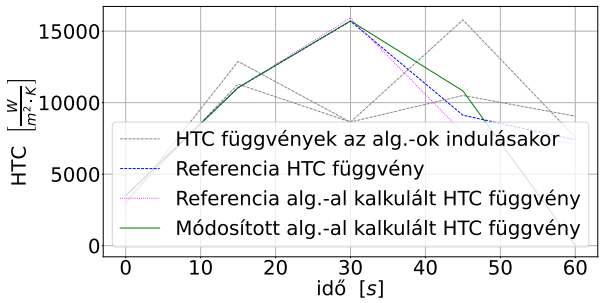


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

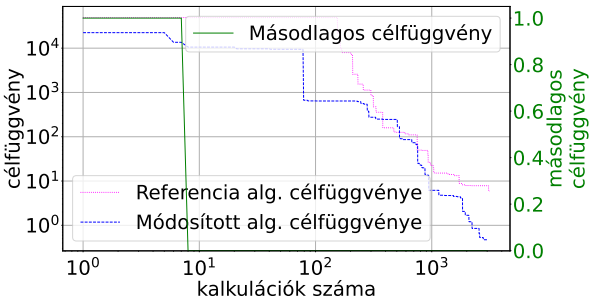


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

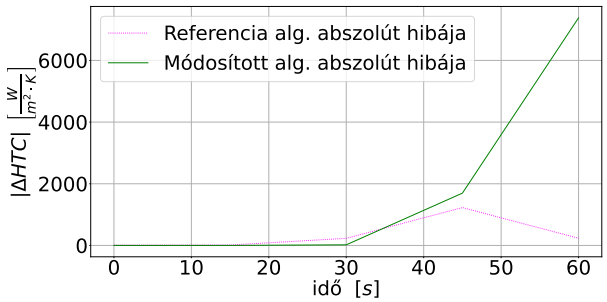
M18. ábra. A cFWA algoritmus futtatásának eredménye 5 dimenziós keresési térben



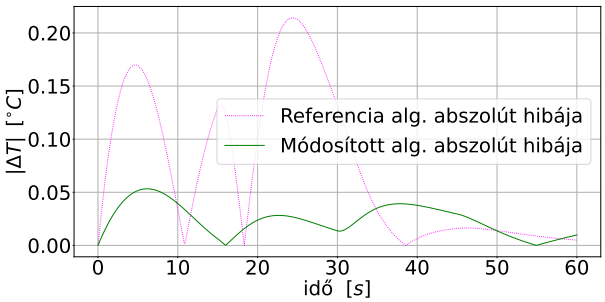
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

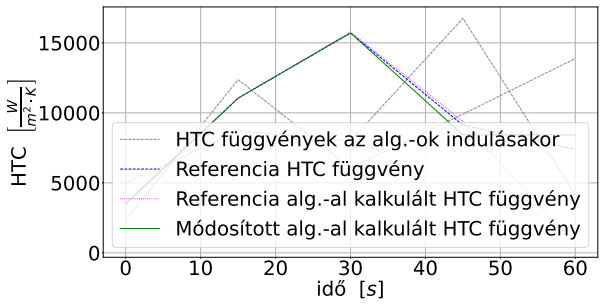


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

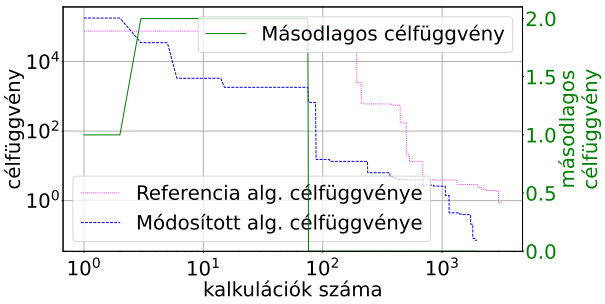


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

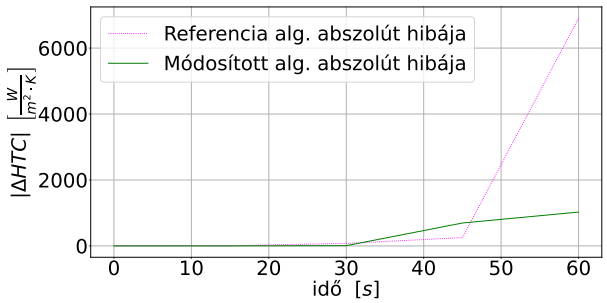
M19. ábra. Az EFWA algoritmus futtatásának eredménye 5 dimenziós keresési térben



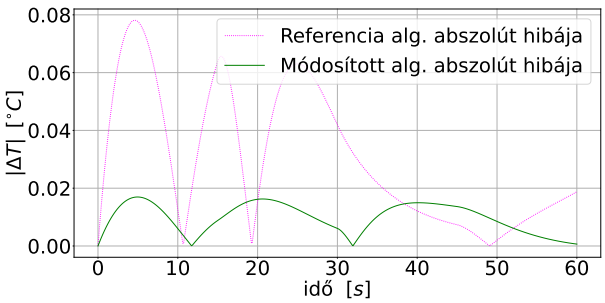
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

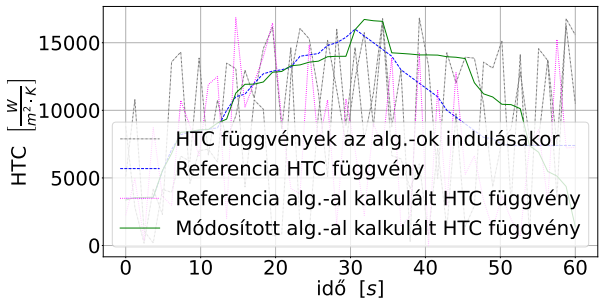


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

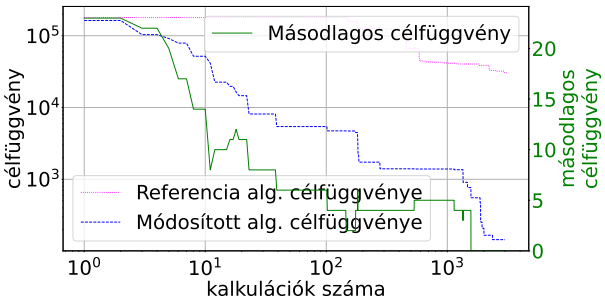


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

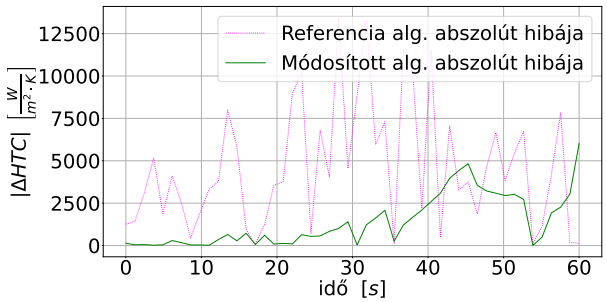
M20. ábra. Az EFWADM algoritmus futtatásának eredménye 5 dimenziós keresési térben



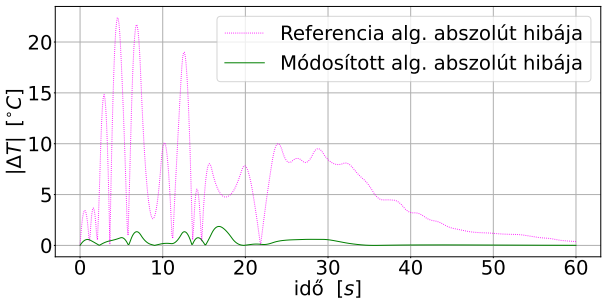
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

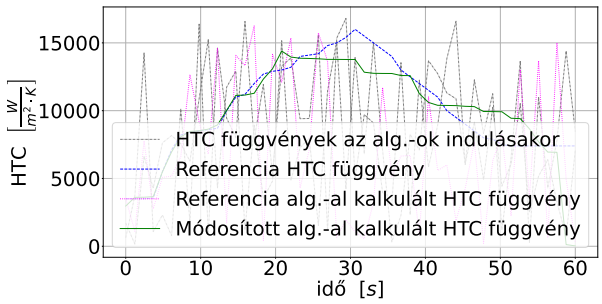


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

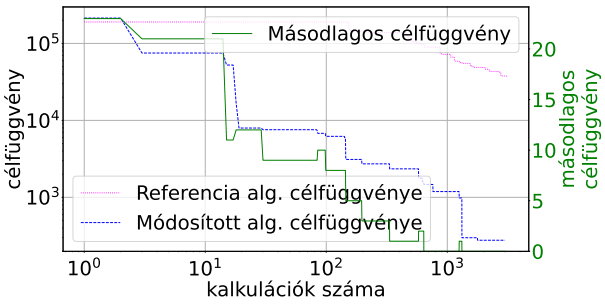


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

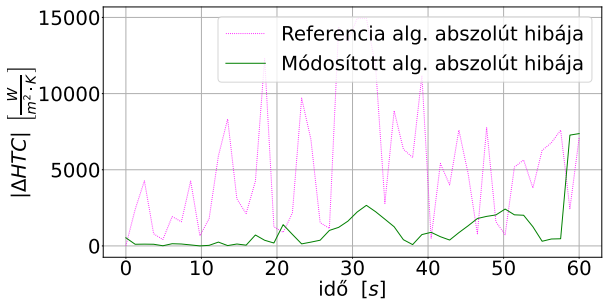
M21. ábra. A FWA algoritmus futtatásának eredménye 50 dimenziós keresési térben



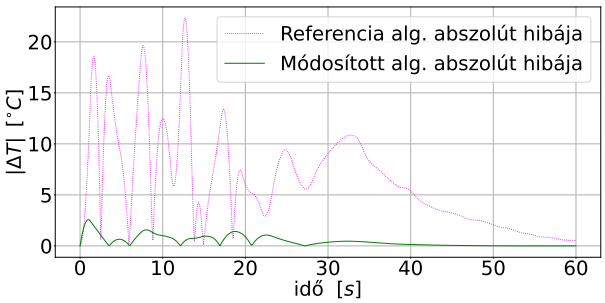
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

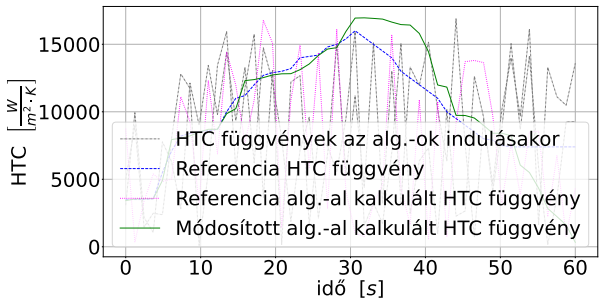


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

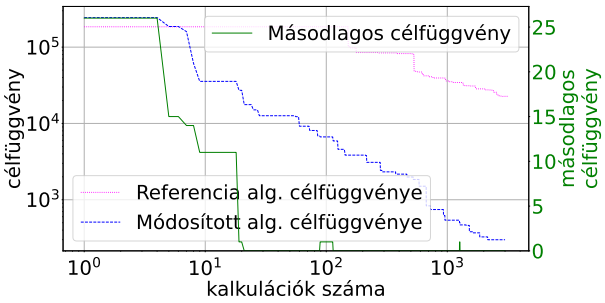


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

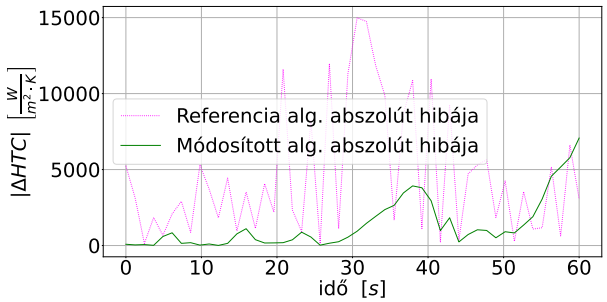
M22. ábra. Az AFWA algoritmus futtatásának eredménye 50 dimenziós keresési térben



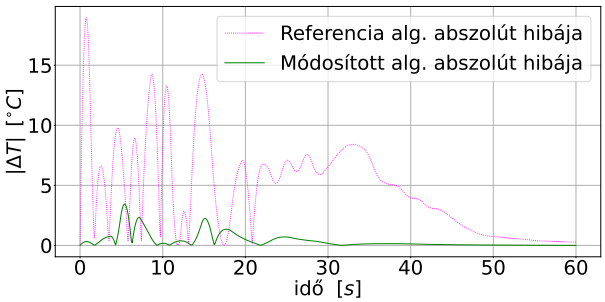
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

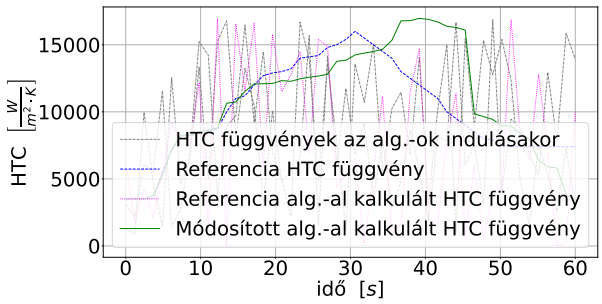


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

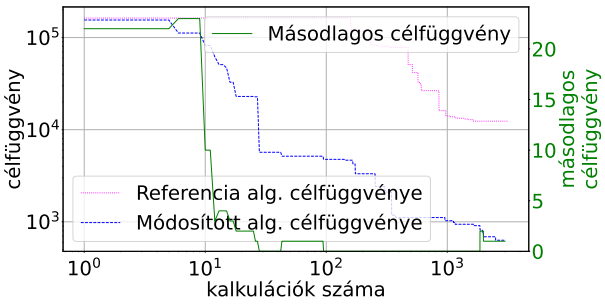


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

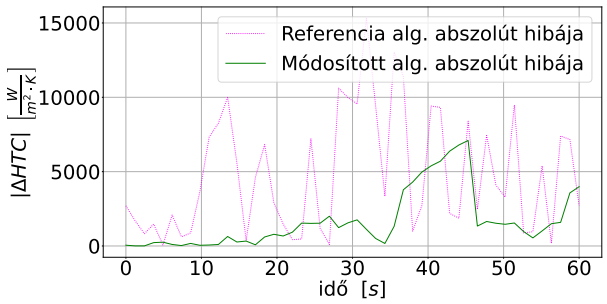
M23. ábra. A cFWA algoritmus futtatásának eredménye 50 dimenziós keresési térben



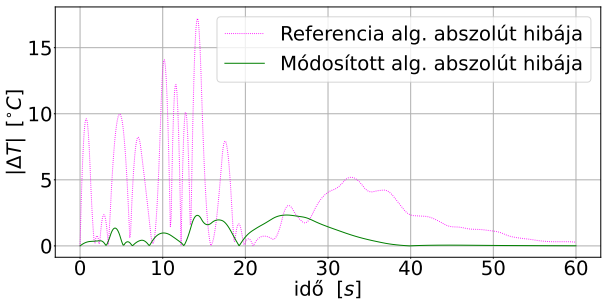
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

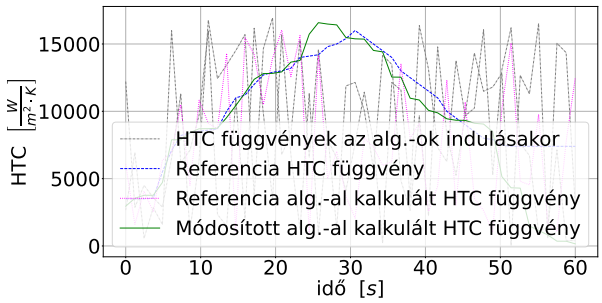


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

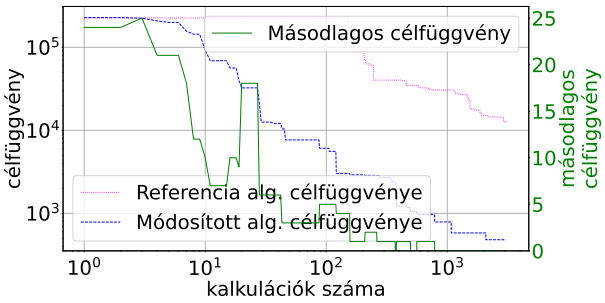


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

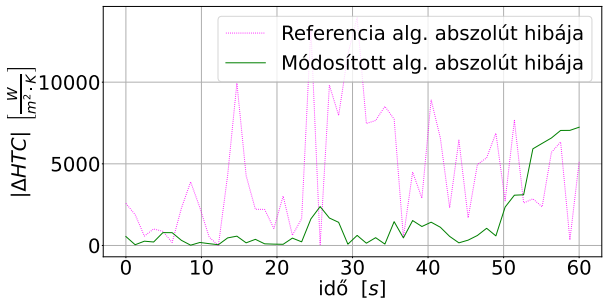
M24. ábra. Az EFWA algoritmus futtatásának eredménye 50 dimenziós keresési térben



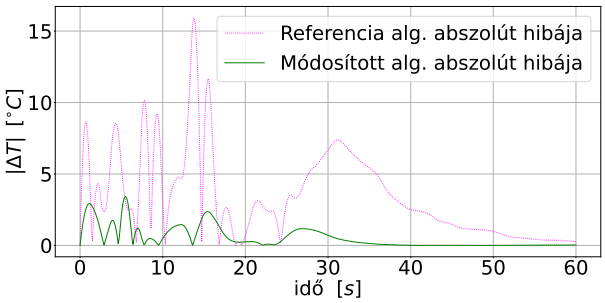
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

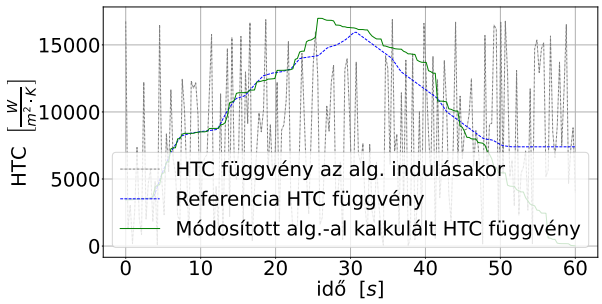


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

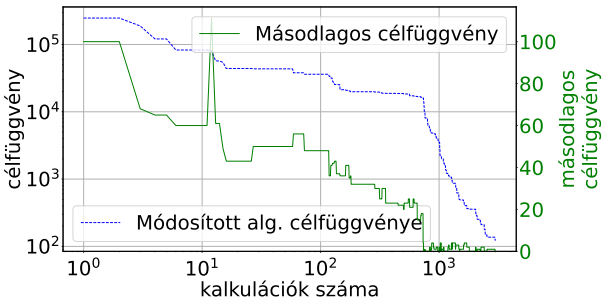


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

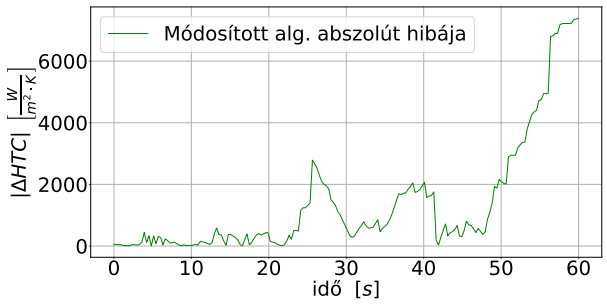
M25. ábra. Az EFWADM algoritmus futtatásának eredménye 50 dimenziós keresési térben



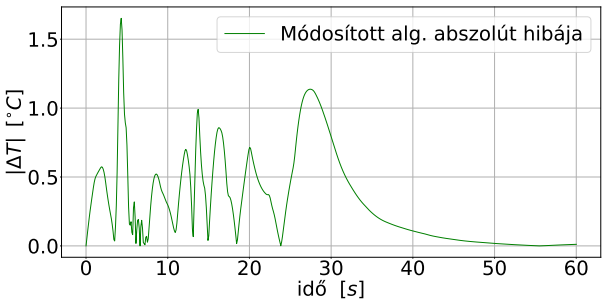
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

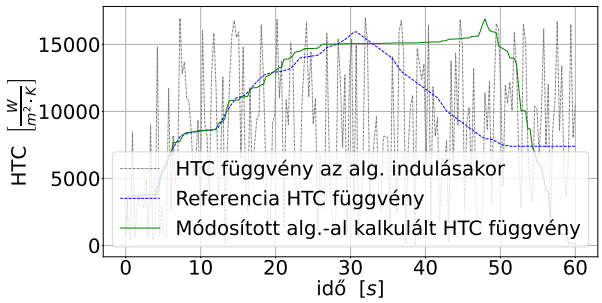


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

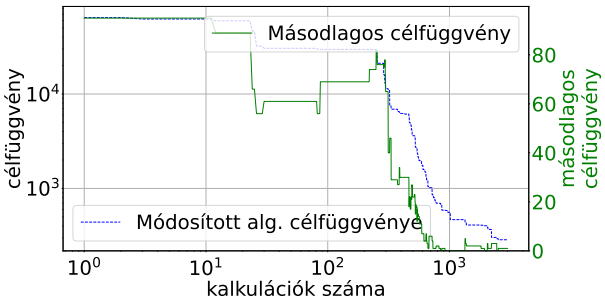


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

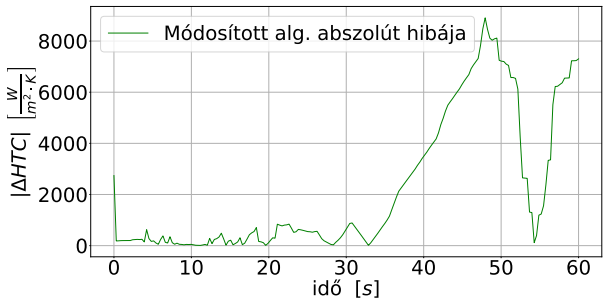
M26. ábra. A FWA algoritmus futtatásának eredménye 200 dimenziós keresési térben



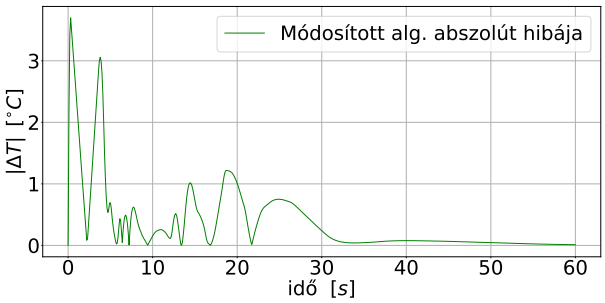
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

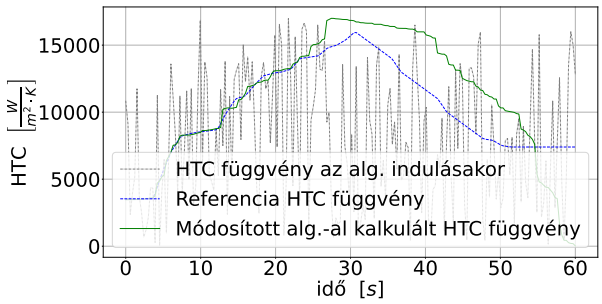


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

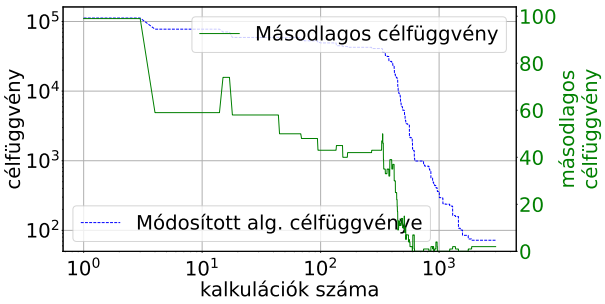


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

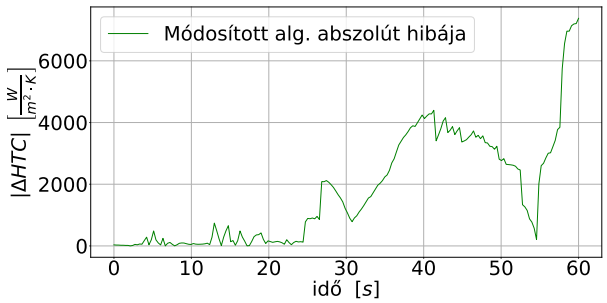
M27. ábra. Az AFWA algoritmus futtatásának eredménye 200 dimenziós keresési térben



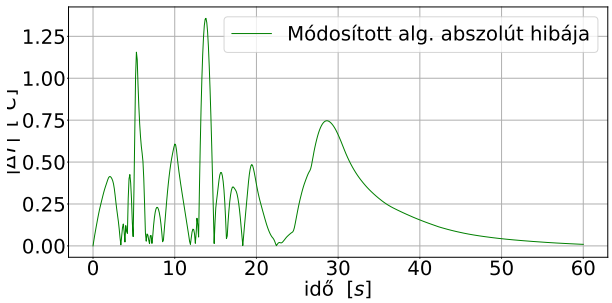
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

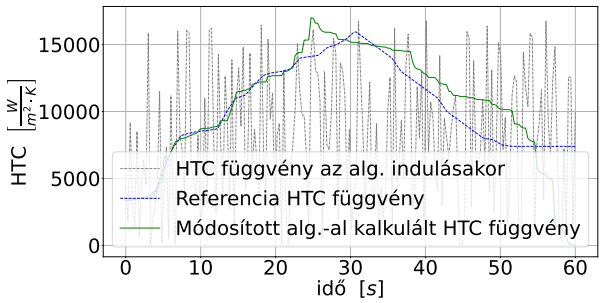


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

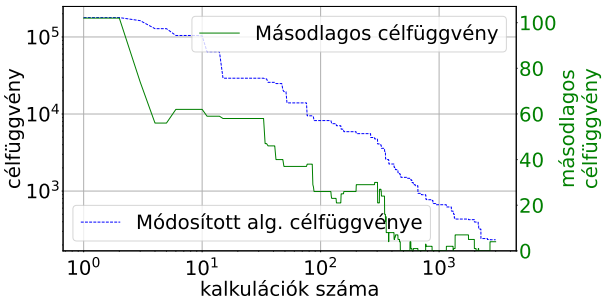


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

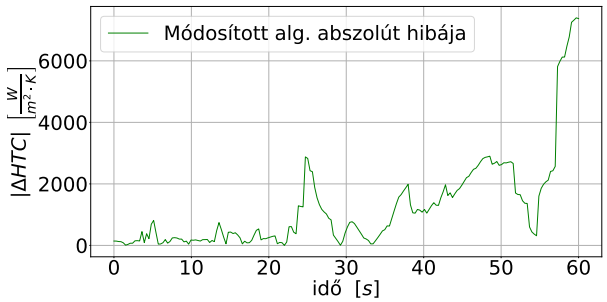
M28. ábra. A cFWA algoritmus futtatásának eredménye 200 dimenziós keresési térben



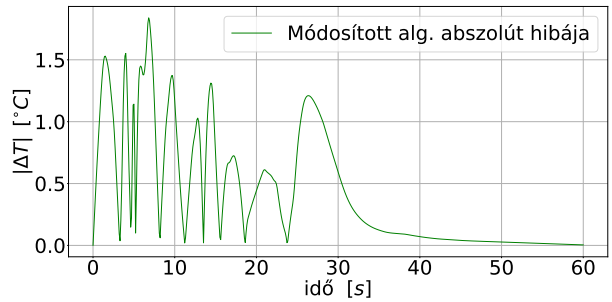
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

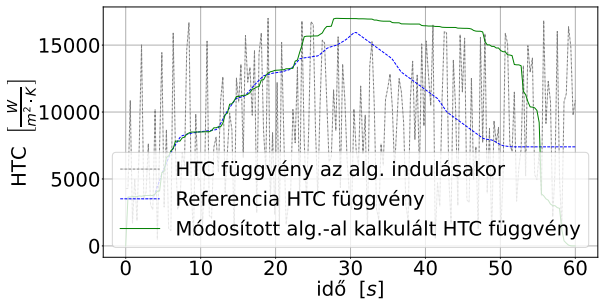


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

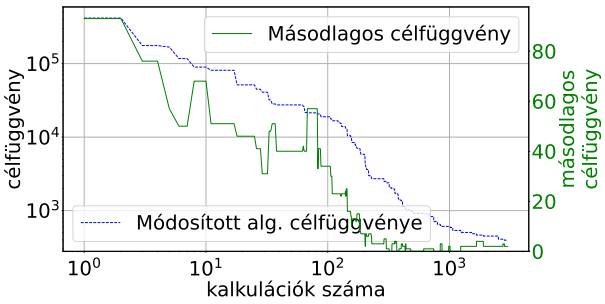


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

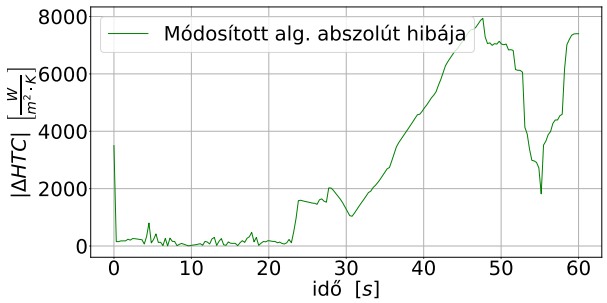
M29. ábra. Az EFWA algoritmus futtatásának eredménye 200 dimenziós keresési térben



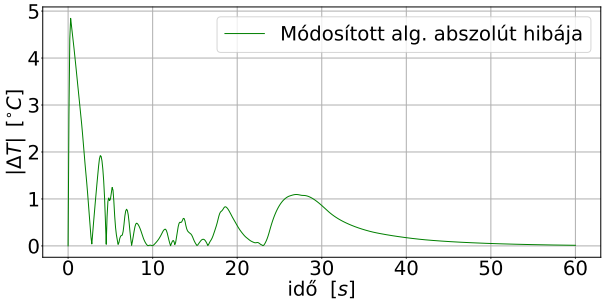
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása



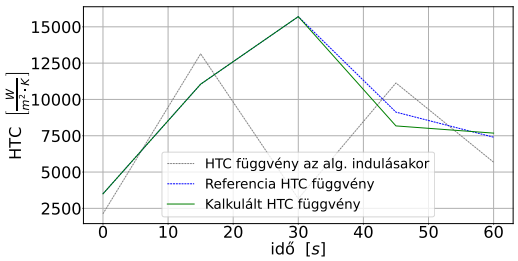
(c) Kalkulált HTC függvényének eltérése a referencia görbétől



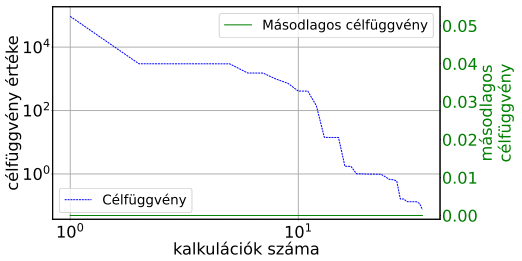
(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

M30. ábra. Az EFWADM algoritmus futtatásának eredménye 200 dimenziós keresési térben

III. Melléklet - Új típusú FWA algoritmus futási eredményei

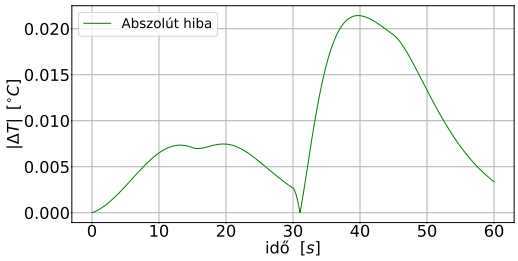


(a) Kalkulált HTC függvény



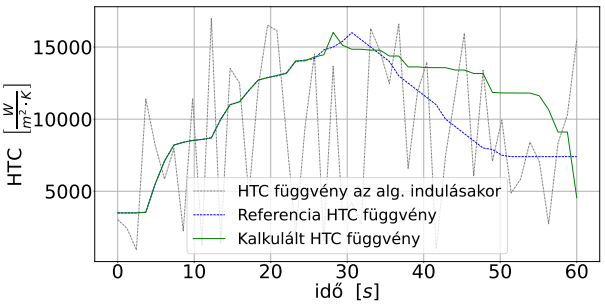
(b) Célfüggvény értékének változása

(c) Kalkulált HTC függvényének eltérése a referencia görbétől

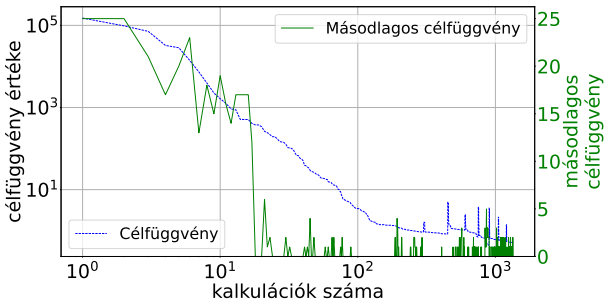


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

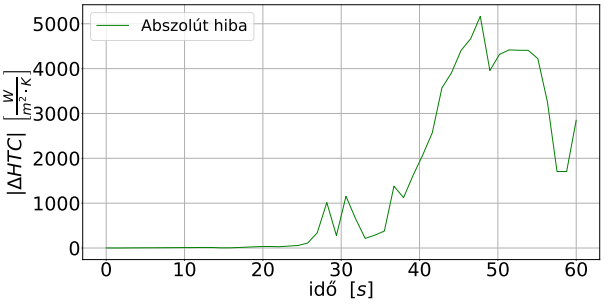
M31. ábra. Új típusú FWA algoritmus futási eredménye 5 dimenziós keresési térben



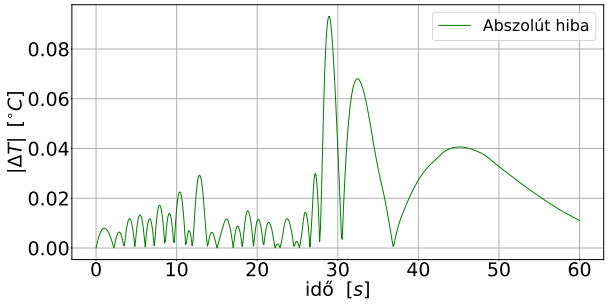
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása

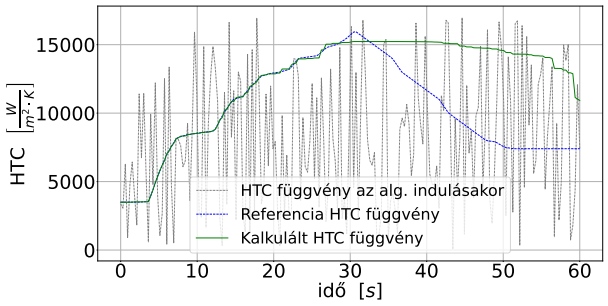


(c) Kalkulált HTC függvényének eltérése a referencia görbétől

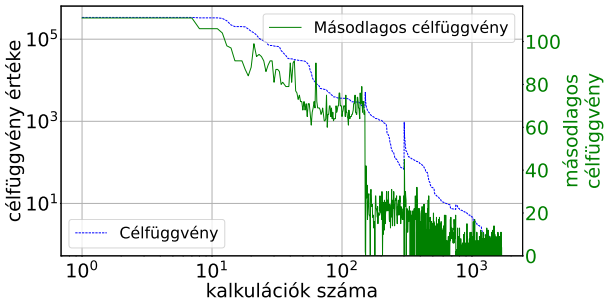


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

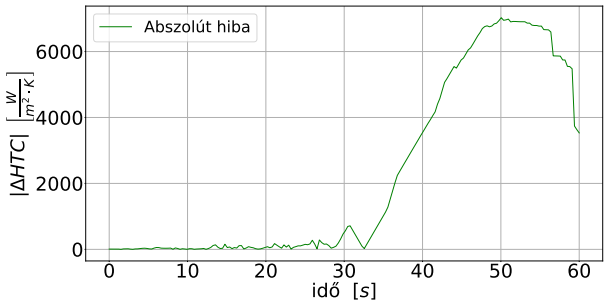
M32. ábra. Új típusú FWA algoritmus futási eredménye 50 dimenziós keresési térben



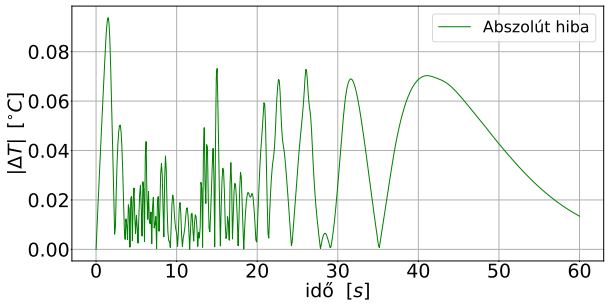
(a) Kalkulált HTC függvény



(b) Célfüggvény értékének változása



(c) Kalkulált HTC függvényének eltérése a referencia görbétől

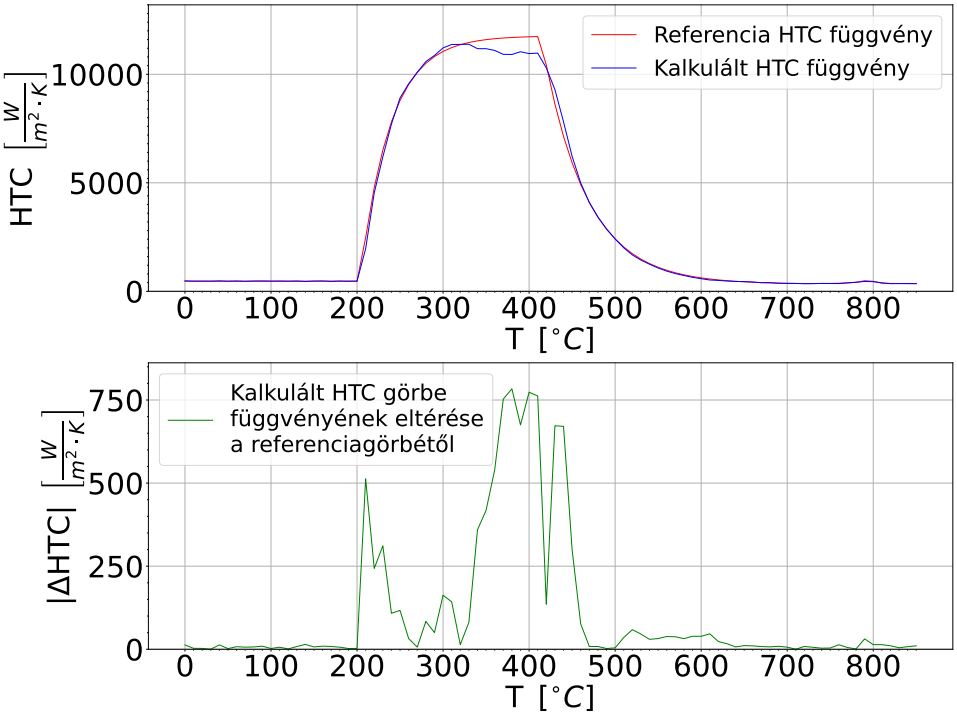


(d) Kalkulált HTC függvényhez tartozó lehűlési görbe eltérése a referencia görbétől

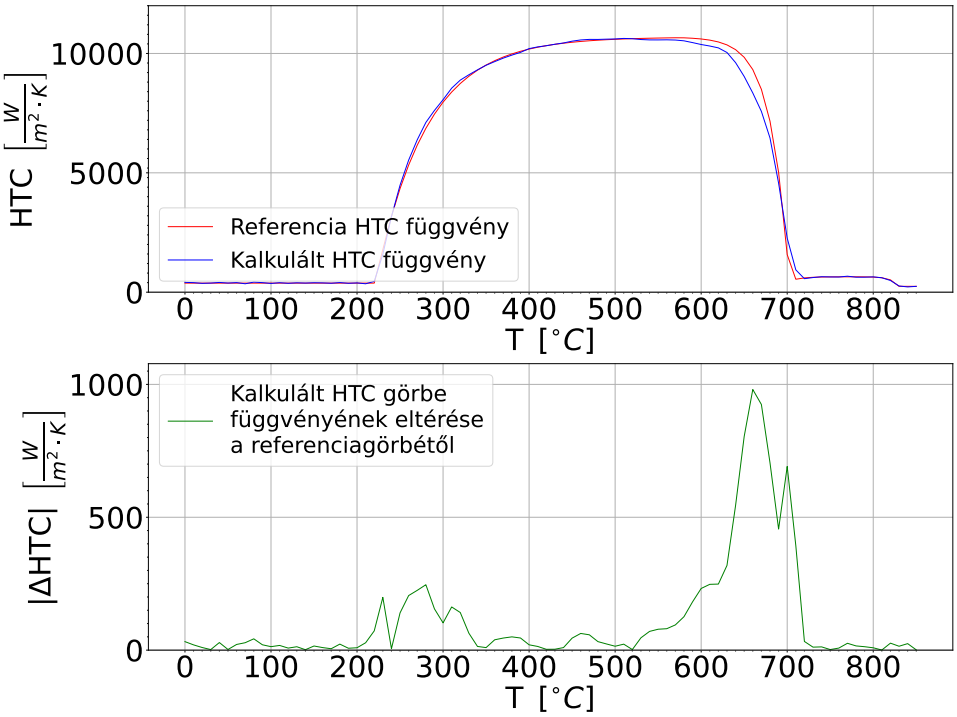
M33. ábra. Új típusú FWA algoritmus futási eredménye 200 dimenziós keresési térben

IV. Melléklet - Neurális hálózat futási eredményei

Néhány kiragadott, a neurális hálózat által generált eredmény és azok összehasonlítása a referencia görbével:

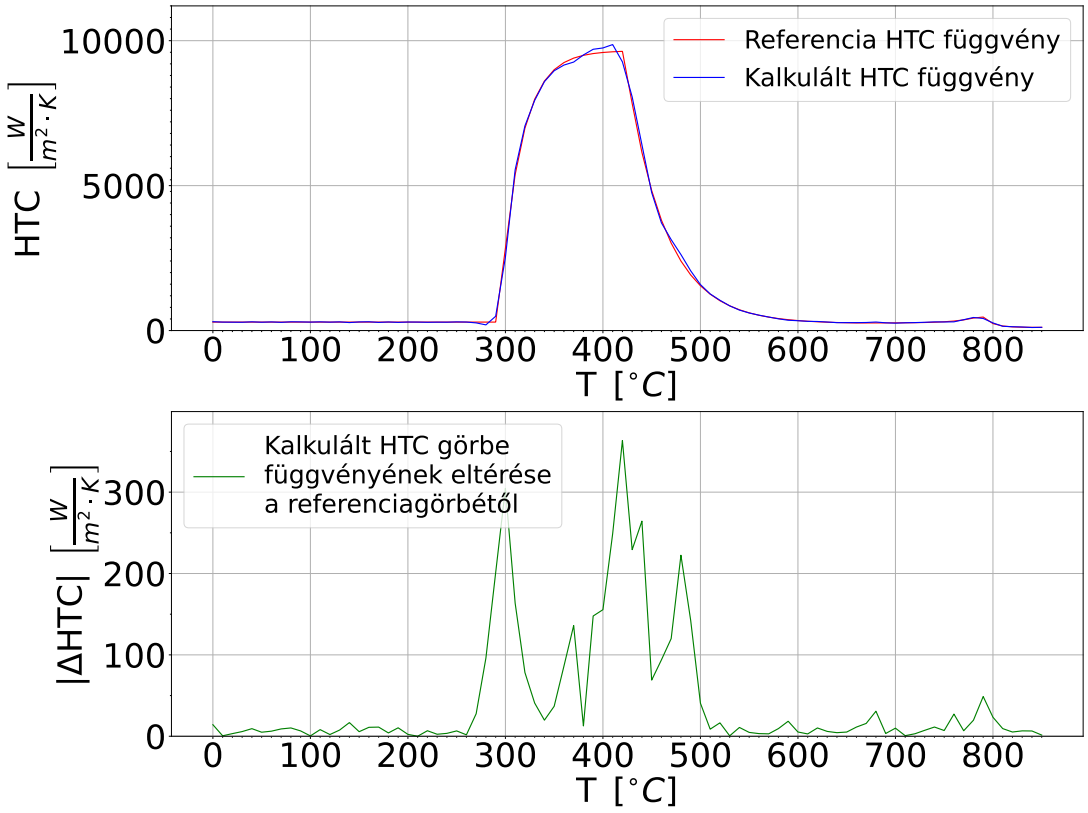


(a)

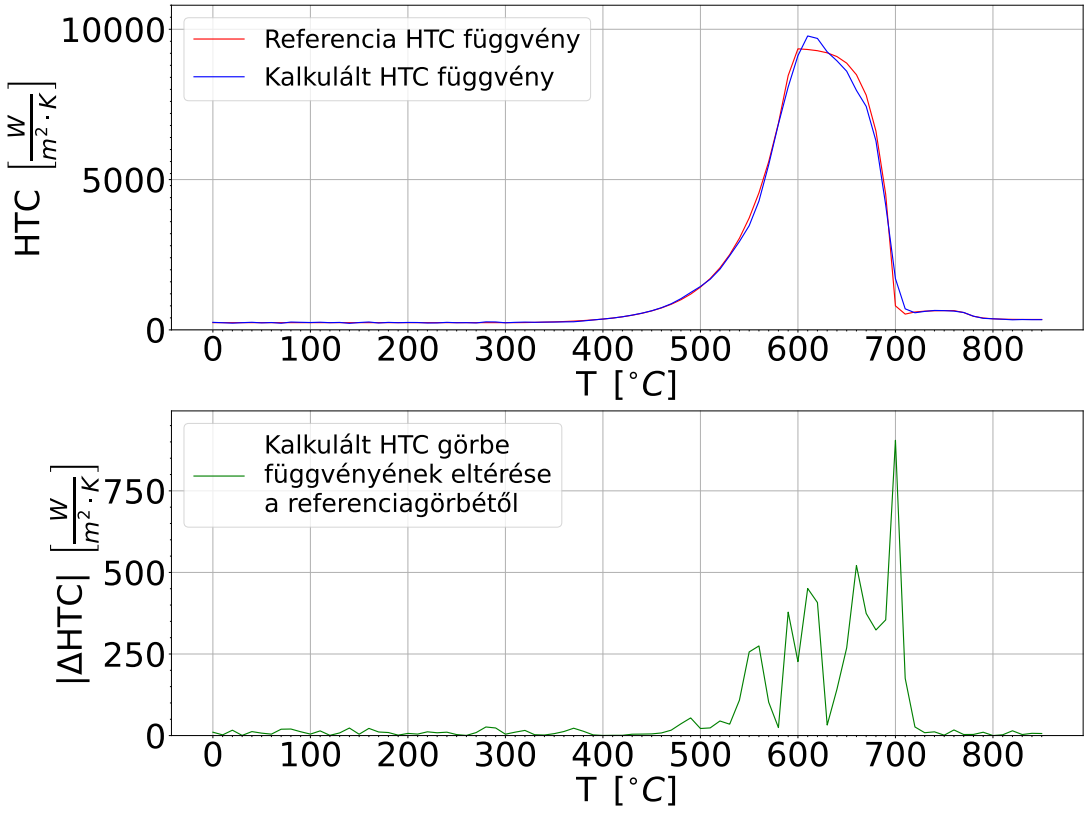


(b)

M34. ábra. Példák a neurális hálózat által generált hőátadási függvényekre, és azoknak a pontossága I.

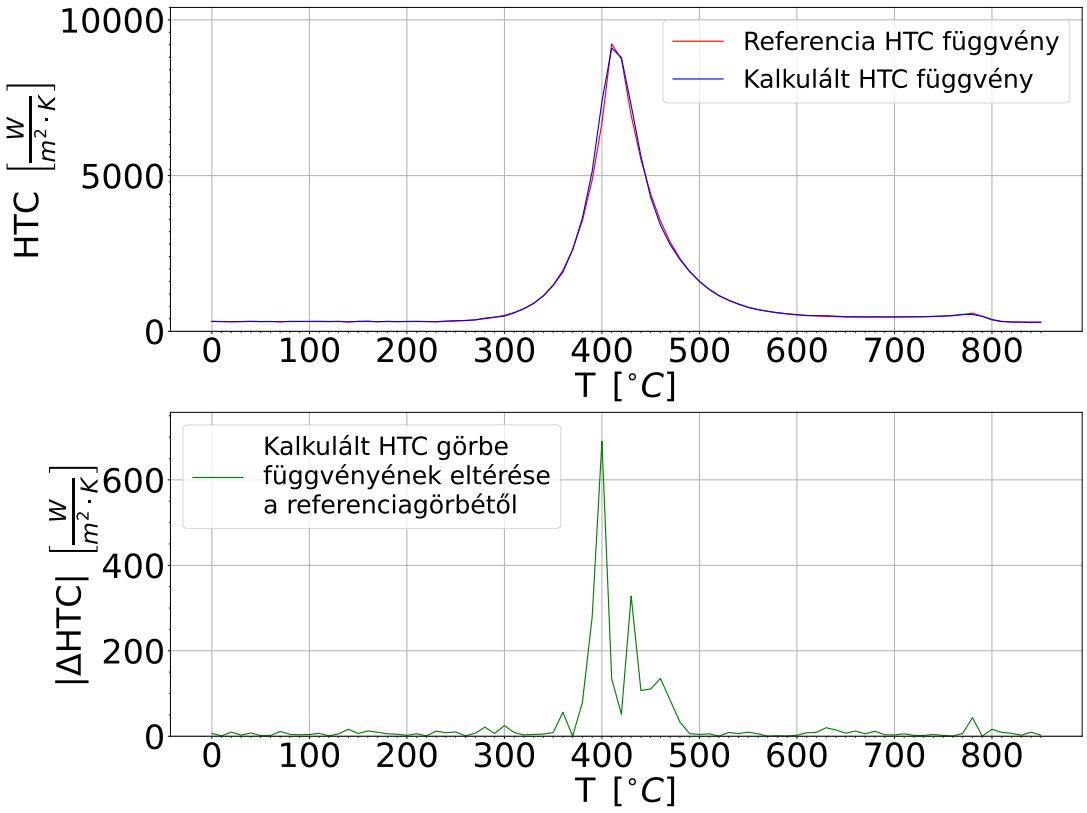


(a)

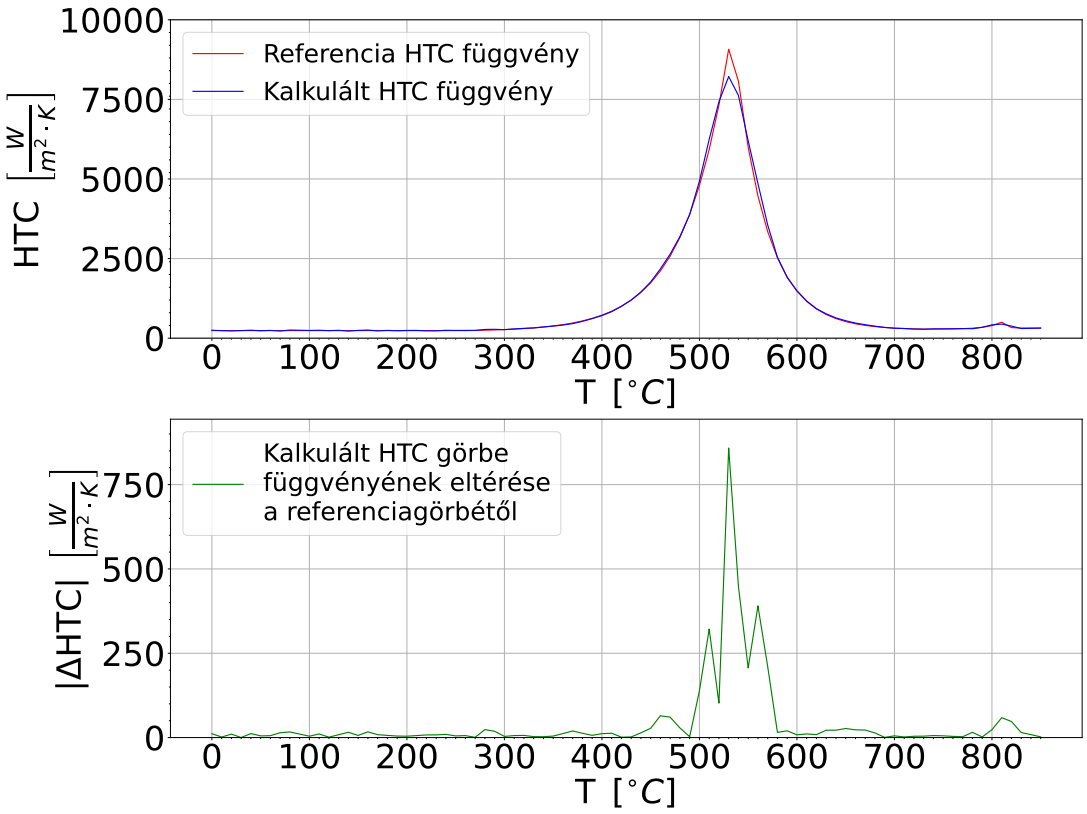


(b)

M35. ábra. Példák a neurális hálózat által generált hőátadási függvényekre, és azoknak a pontossága II.



(a)



(b)

M36. ábra. Példák a neurális hálózat által generált hőátadási függvényekre, és azoknak a pontossága III.

V. Melléklet - Bio-inspirált algoritmusok futási eredményeinek számszerű adatai

Algoritmus	Dim.	Futási idő csak CPU*-val (sec)			
		min.	max.	átlag	medián
Másodlagos célfüggvény és az új mapping operator használatával					
PSO	5	27,73	100,46	68,74	76,06
PSO-Co	5	1,61	97,59	22,24	2,56
PSO-In	5	2,10	96,33	48,46	46,13
QPSOT1	5	2,08	97,74	38,00	3,40
QPSOT2	5	95,50	97,38	96,60	96,81
PSO	50	140,83	204,94	181,20	191,15
PSO-Co	50	176,50	199,57	190,38	193,48
PSO-In	50	194,05	198,33	196,70	197,33
QPSOT1	50	193,10	196,70	194,20	193,10
QPSOT2	50	125,06	194,57	173,40	193,78
PSO	200	826,90	852,68	839,67	841,08
PSO-Co	200	838,58	841,28	839,90	839,40
PSO-In	200	836,97	841,52	839,64	840,33
QPSOT1	200	831,28	843,39	834,07	831,28
QPSOT2	200	833,45	836,08	834,70	834,76
Másodlagos célfüggvény és az új mapping operátor használata nélkül					
PSO	5	1,45	2,88	1,80	1,45
PSO-Co	5	2,63	25,61	9,30	2,63
PSO-In	5	1,58	2,91	2,06	1,78
QPSOT1	5	0,75	1,77	1,23	1,19
QPSOT2	5	0,98	1,58	1,25	1,20
PSO	50	199,40	219,29	205,99	199,40
PSO-Co	50	201,56	203,45	202,86	203,40
PSO-In	50	202,33	203,52	202,81	202,58
QPSOT1	50	166,09	203,32	191,16	200,22
QPSOT2	50	105,92	146,31	120,50	106,89

Algoritmus	Dim.	Futási idő CPU*-val (sec)			
		min.	max.	átlag	medián
FWA	5	31,97	36,22	33,33	31,97
AFWA	5	29,45	31,82	29,97	29,50
cFWA	5	32,29	34,68	32,81	32,31
EFWA	5	30,06	33,77	30,89	30,08
EFWADM	5	28,19	31,29	28,85	28,27
FWA	50	413,70	428,04	419,1	413,70
AFWA	50	407,78	423,53	411,47	408,35
cFWA	50	402,22	410,43	405,09	403,82
EFWA	50	403,24	406,84	404,35	403,47
EFWADM	50	402,27	405,94	404,16	404,01
FWA	200	2520,70	2614,17	2583,61	2607,54
AFWA	200	2531,39	2566,79	2551,42	2553,20
cFWA	200	2485,80	2567,86	2526,29	2528,52
EFWA	200	2511,24	2554,18	2525,74	2511,24
EFWADM	200	2511,96	2528,37	2517,80	2511,96
FWA	5	89,24	206,02	169,98	203,02
AFWA	5	196,71	203,00	200,70	202,00
cFWA	5	194,54	199,78	197,57	198,50
EFWA	5	195,03	198,04	196,76	197,08
EFWADM	5	195,53	197,35	196,31	196,31
FWA	50	416,91	433,31	424,30	423,80
AFWA	50	406,98	415,72	411,65	412,72
cFWA	50	404,79	406,60	405,80	406,00
EFWA	50	404,68	407,89	406,69	407,10
EFWADM	50	404,61	408,37	406,46	406,28

M1. táblázat. PSO és FWA algoritmusok futási eredményei 5, 50 és 200 dimenziókban különböző konfigurációkban csak CPU* használata esetében (*15 core)

Algoritmus	Dim.	Kalkuláció		Célfüggvény						Másodlagos	futási idő CPU*-val és GPU-val (sec)			
		min.	max.	min.	max.	átlag	median	std.dev.	std.err.	célfüggvény	min.	max.	átlag	medián
Másodlagos célfüggvény és az új mapping operator használatával														
PSO	5	43150	150050	0,096	2337,11	564,56	39,43	900,25	402,60	0	8,30	33,40	25,82	33,26
PSO-Co	5	2400	150050	0,021	36,07	7,27	0,08	14,40	6,44	0	0,51	30,53	6,65	0,71
PSO-In	5	3150	150050	0,063	181,37	36,34	0,09	72,52	32,43	0	0,86	29,26	14,83	12,50
QPSOT1	5	3400	150050	0,063	2358,76	471,90	0,09	943,43	421,91	0	0,66	30,67	11,87	1,08
QPSOT2	5	150050		0,516	2497,50	833,49	5,30	1051,06	470,05	0	28,43	30,31	29,50	29,76
PSO	50	210200	300200	0,063	1,85	0,61	0,30	0,65	0,29	0	6,01	69,28	49,80	67,90
PSO-Co	50	270200	300200	0,047	0,46	0,29	0,31	0,16	0,07	0	5,88	63,90	46,32	63,52
PSO-In	50	193400	300200	0,168	0,54	0,26	0,19	0,14	0,06	0	8,39	62,67	46,36	62,60
QPSOT1	50	186200	300200	0,119	1,15	0,66	0,56	0,42	0,19	0	7,44	61,03	43,45	57,44
QPSOT2	50	195200	300200	0,072	2,12	0,62	0,25	0,76	0,34	0	7,67	58,90	43,15	57,66
PSO	200	1200400		3,22	200058,00	85721,20	65522,00	71577,67	32010,51	0-84	450,33	476,11	463,10	464,51
PSO-Co	200	1200400		1,09	206258,00	41253,70	1,76	82502,15	36896,08	0	462,01	464,71	463,33	462,83
PSO-In	200	1200400		1,01	389372,00	77880,10	1,88	155745,95	69651,71	0	460,40	464,95	463,07	463,36
QPSOT1	200	1200400		1,22	47,31	21,12	19,06	15,83	7,08	0	454,71	466,82	458,50	454,71
QPSOT2	200	1200400		4,16	90,77	24,91	5,89	33,36	14,92	0	456,88	459,51	458,20	458,19
Másodlagos célfüggvény és az új mapping operátor használata nélkül														
PSO	5	2300	4550	0,06	0,09	0,07	0,07	0,01	0,01	-	0,44	0,85	0,60	0,44
PSO-Co	5	4050	39750	0,03	0,09	0,07	0,08	0,02	0,01	-	0,77	8,41	2,99	0,77
PSO-In	5	2500	4600	0,04	0,09	0,07	0,07	0,02	0,01	-	0,58	0,87	0,69	0,62
QPSOT1	5	1150	2650	0,05	0,09	0,07	0,07	0,02	0,01	-	0,24	0,66	0,44	0,42
QPSOT2	5	1500	2400	0,04	0,08	0,06	0,06	0,01	0,01	-	0,31	0,48	0,39	0,39
PSO	50	300200		1908.29	5552,61	3095,81	2674,14	1286,32	575,26	-	63,73	83,63	70,33	63,73
PSO-Co	50	300200		649,33	2866,48	1562,84	1306,44	793,16	354,71	-	65,89	67,79	67,11	67,54
PSO-In	50	300200		876,08	4945,00	3008,22	3266,82	1414,54	632,60	-	66,67	67,86	67,15	66,92
QPSOT1	50	244600	300200	0,48	200678,00	40139,11	5,28	80269,45	35897,59	-	54,76	67,65	62,88	65,65
QPSOT2	50	158100	217000	0,48	0,49	0,48	0,48	0,00	0,00	-	35,30	49,17	40,21	35,63

M2. táblázat. PSO algoritmusok futási eredményei 5, 50 és 200 dimenziókban különböző konfigurációkban CPU*-val és GPU-val (*15 core)

Algoritmus	Dim.	Kalkuláció min.	Kalkuláció max.	Célfüggvény min.	Célfüggvény max.	Célfüggvény átlag	Célfüggvény median	Célfüggvény std.dev.	Célfüggvény std.err.	Másodlagos célfüggvény	Futási idő CPU*-val és GPU-val (sec) min.	Futási idő CPU*-val és GPU-val (sec) max.	Futási idő CPU*-val és GPU-val (sec) átlag	Futási idő CPU*-val és GPU-val (sec) medián
Másodlagos célfüggvény és az új mapping operator használatával														
FWA	5	210200	300200	0,063	1,85	0,61	0,31	0,65	0,29	0	46,01	69,28	61,23	65,90
AFWA	5	270200	300200	0,047	0,46	0,29	0,31	0,16	0,07	0	46,77	63,90	57,91	61,09
cFWA	5	300200		0,168	0,54	0,26	0,19	0,14	0,06	0	58,39	62,67	60,53	60,67
EFWA	5	300200		0,119	1,15	0,66	0,56	0,42	0,19	0	57,44	61,03	58,81	57,44
EFWADM	5	195200	300200	0,072	2,12	0,62	0,25	0,76	0,34	0	37,67	58,90	52,43	58,66
FWA	50	600400		143,93	449,59	271,83	245,95	107,09	47,89	0-3	142,67	157,01	148,07	142,67
AFWA	50	600400		86,55	312,65	235,06	290,01	86,28	38,59	0	136,74	152,50	140,43	137,33
cFWA	50	600400		152,60	364,27	265,80	277,04	75,10	33,59	2	131,19	139,40	134,05	132,84
EFWA	50	600400		223,14	468,48	346,33	323,60	95,18	42,56	0-1	132,21	135,81	133,31	132,44
EFWADM	50	600400		97,31	443,20	272,68	260,95	127,16	56,87	0-3	131,24	134,91	132,85	132,98
FWA	200	3601800		79,64	123,37	106,94	113,86	107,09	47,89	0	838,02	931,49	900,93	924,86
AFWA	200	3601800		68,77	292,92	169,19	145,97	86,28	38,59	0	848,71	884,11	869,34	870,52
cFWA	200	3601800		75,92	162,06	105,99	90,99	75,10	33,59	2-4	803,12	885,18	843,61	845,84
EFWA	200	3601800		64,63	131,48	94,31	83,88	95,18	42,56	0	828,56	871,50	843,06	828,56
EFWADM	200	3601800		83,22	395,51	165,61	107,31	127,16	56,87	2	829,28	845,69	835,21	829,28
Másodlagos célfüggvény és az új mapping operátor használata nélkül														
FWA	5	135200	300200	0,057	0,69	0,33	0,29	0,22	0,10	-	29,41	70,35	55,56	64,35
AFWA	5	300200		0,131	2,43	0,67	0,19	0,89	0,40	-	61,05	67,34	64,55	64,34
cFWA	5	300200		0,287	2,21	0,98	0,41	0,79	0,36	-	58,88	64,11	62,22	63,11
EFWA	5	300200		0,112	1,95	0,63	0,36	0,67	0,30	-	59,36	62,37	61,10	61,42
EFWADM	5	300200		0,250	0,85	0,50	0,42	0,22	0,10	-	59,87	61,68	60,62	60,64
FWA	50	600400		15322,30	30505,90	19485,60	16769,30	5629,95	2517,79	-	145,88	162,28	153,27	152,77
AFWA	50	600400		11800,70	29080,80	18159,70	15871,10	6129,34	2741,12	-	135,94	144,69	141,42	142,69
cFWA	50	600400		12275,20	34094,50	21955,00	21644,00	7405,68	3311,92	-	133,76	135,57	134,96	135,37
EFWA	50	600400		12375,00	35614,40	20573,80	18862,30	8011,91	3583,03	-	133,65	136,86	135,67	136,13
EFWADM	50	600400		12425,40	19368,40	16359,60	17208,40	2459,81	1100,06	-	133,58	137,34	135,43	135,25

M3. táblázat. FWA algoritmusok futási eredményei 5, 50 és 200 dimenziókban különböző konfigurációkban CPU*-val és GPU-val (*15 core)

VI. Melléklet - A szerző témához kapcsolódó publikációi

1. Zoltán Fried és tsai.: „On the Nature-Inspired Algorithms Applied to Characterize Heat Transfer Coefficients”. *Thermal Processing in Motion*. Spartanburg: ASM, 2018, 47–51. old.
2. I. Felde, Z. Fried és S. Szénási: „Solution of 2-D Inverse Heat Conduction Problem with Graphic Accelerator”. *Materials Performance and Characterization* 6.5 (2017), 882–893. old. ISSN: 2379–1365. DOI: 10.1520/mpc20170008.
3. Zoltán Fried, Sándor Szénási és Imre Felde: „Reconstruction of a heat transfer coefficients by using FWA approach”. Budapest, Hungary. Budapest, Hungary: IEEE, 2018. nov., 99–000104. old. ISBN: 978-1-7281-1118-6. DOI: 10.1109/CINTI.2018.8928227.
4. Zoltán Fried, Sándor Szénási és Imre Felde: „Prediction of objective function value for heat transfer coefficient function reconstruction by FWA”. *2019 IEEE 13th International Symposium on Applied Computational Intelligence and Informatics (SACI)*. Timisoara, Romania: IEEE, 2019. máj., 305–308. old. ISBN: 978-1-7281-0685-4. DOI: 10.1109/SACI46893.2019.9111623.
5. Z Fried, I Felde és S Szénási: „Enhancing the Firework Algorithm ecosystem for the reconstruction of the HTC function”. *IOP Conference Series: Materials Science and Engineering* 903 (2020. aug.), 12020. old. ISSN: 1757-8981. DOI: 10.1088/1757-899x/903/1/012020.
6. Zoltán Fried, Imre Felde és József K. Tar: „On the Simulation of Cooling Curves Using Simple Functional Formats”. *Acta Polytechnica Hungarica* 17.9 (2020), 109–124. old. ISSN: 1785-8860. DOI: 10.12700/aph.17.9.2020.9.6.
7. Imre Felde és tsai.: „Investigation of parallelized PSO algorithm applied to estimate complex HTC”. *Proceedings of the 5th Asian Conference on heat Treatment and Surface Engineering*. 2016, 263–270. old.
8. Zoltán Fried és tsai.: „Komplex hőátadási együttható rekonstrukciója bio-inspirált módszer alkalmazásával”. *XXVII. Hőkezelő és anyagtudomány a gépgyártásban országos konferencia és szakkiállítás külföldi részvevőkkel*. 2016, 53–58. old.
9. Zoltán Fried és tsai.: „Prediction of thermal boundary conditions by using FWA”. *Proceedings of the International Conference on Quenching and Distortion Engineering (QDE) 2018*. 2018.
10. Zoltán Fried, Imre Felde és Sándor Szénási: „Komplex hőátadási együttható rekonstrukciója az FWA algoritmus alkalmazásával”. *XXVIII. Hőkezelő és anyagtudomány a gépgyártásban országos konferencia és szakkiállítás külföldi résztvevőkkel*. 2019, 274–279. old.

11. Zoltán Fried és tsai.: „Parallelized Particle Swarm Optimization to Estimate the Heat Transfer Coefficients of Palm Oil, Canola Oil, Conventional, and Accelerated Petroleum Oil Quenchants”. *Materials Performance and Characterization* 8 (2018), 96–113. old. ISSN: 2379-1365. DOI: 10.1520/MPC20180049.

12. Sandor Szenasi, Zoltan Fried és Imre Felde: „Training of Artificial Neural Network to Solve the Inverse Heat Conduction Problem”. *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE, 2020. jan., 293–298. old. DOI: 10.1109/SAMI48414.2020.9108733.

13. Sandor Szenasi, Zoltan Fried és Imre Felde: „GPU Accelerated Heat Transfer Simulation Supporting Heuristics To Solve The Inverse Heat Conduction Problem”. *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*. IEEE, 2020. jan., 287–292. old. DOI: 10.1109/sami48414.2020.9108768.

14. Zoltán Fried, Sándor Szénási és Imre Felde: „Reconstruction of the heat transfer coefficients by using hybrid (FWA + gradient) approach”. *IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI 2020)*. 2020, 299–304. old. DOI: 10.1109/SAMI48414.2020.9108767.

15. Imre Felde, Zoltán Fried és Sándor Szénási: „Application of Nature-Inspired Algorithms to Solve Inverse Heat Conduction Problems”. *24th IFHTSE CONGRESS 2017*. 2017.