Univerza *v Ljubljani*
Fakulteta za *matematiko in fiziko*

Izračun Airyjevih funkcij

1. naloga pri Matematično-fizikalnem praktikumu

Avtor: Marko Urbanč (28191096)
Predavatelj: prof. dr. Borut Paul Krševan

12.10.2021

Kazalo

1 Uvod **2**

2 Naloga **3**

3 Opis reševanja **3**

3.1 Maclaurinova vrsta 3

3.2 Asimptotski vrsti 4

4 Rezultati **5**

4.1 Funkcija Ai 5

4.2 Funkcija Bi 7

4.3 Natančnost `mpmath.airy` 9

4.4 Komentarji in možne izboljšave 10

Literatura **11**

1 Uvod

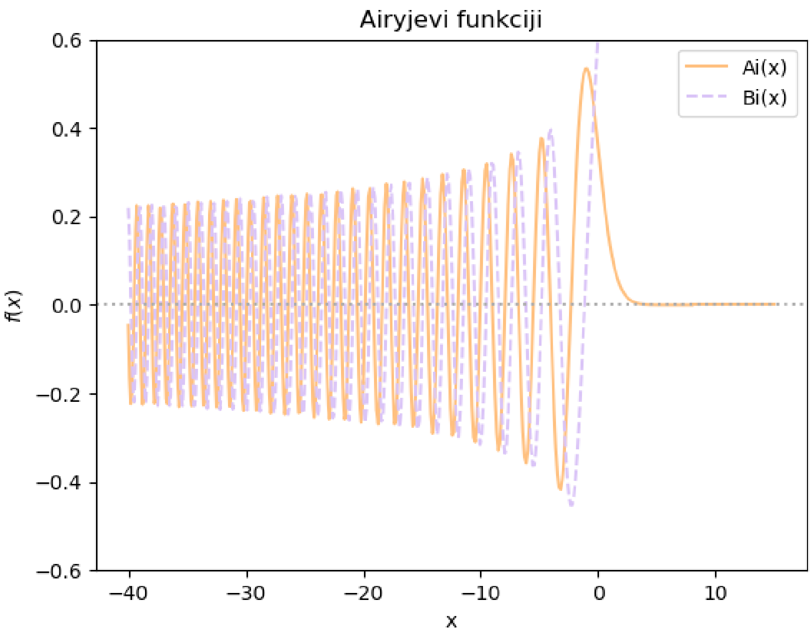
Airyjevi funkciji $Ai(x)$ in $Bi(x)$ sta linearno nedovisni rešitvi *Stoksove* oz. *Airyjeve enačbe*, ki je matematično gledano linearna diferencialna enačba drugega reda.

$$\frac{d^2y}{dx^2} - xy = 0$$

Rešitvi lahko predstavimo v integralski obliki kot

$$\mathrm{Ai}(x) = \frac{1}{\pi}\int_0^\infty \cos(t^3/3 + xt)t\,, \quad \mathrm{Bi}(x) = \frac{1}{\pi}\int_0^\infty \left[\mathrm{e}^{-t^3/3+xt} + \sin(t^3/3 + xt)\right] t\,.$$

Enačba je zanimva, ker je to najpreprostejša diferencialna enačba drugega reda, ki ima obračališče. Obračališče je točka, kjer se rešitve enačbe spremenijo iz oscilirajoče v eksponentno. Enačba je poimenovana po angleškem astronomu in fiziku Airyju, ki jo je razvil, ko je raziskoval pojave v optiki. Airyjevi funkciji sta zanimivi tudi za kvantno mehaniko saj rešita nestacionarno Schrödingerjevo enačbo za delec v trikotni potencialni jami.



Slika 1: Graf Airyjevih funkcij za realne argumente

2 Naloga

Z uporabo kombinacije Maclaurinove vrste in asimptotskega razvoja moramo poiskati čim učinkovitejši postopek za izračun vrednosti Ariyjevih funkcij na vsej realni osi. Želimo imeti absolutno napako manjšo od 10^{-10}. Enako poskusimo z relativno napako in preverimo, ali je sploh dosegljiva natančnost manjša od 10^{-10}.

3 Opis reševanja

Problema sem se lotil v Pythonu. Moja ideja reševanja je bila sledeča. Želel sem napisati osnove in delujočo verzijo z uporabo knjižnice `NumPy` in nato ustrezno kodo popraviti, da uporablja funkcije iz programskega paketa `mpmath`. Prva verzija programa je precej dobro delovala, a sem vseeno želel računati s poljubno natančnostjo. Izkazalo pa se je, da funkcije paketa `mpmath` ne zmorejo računati z *ndarray*-i, ki jih uporablja `NumPy`. Program sem torej znova napisal z uporabo matematičnih funkcij `mpmath` in preprosto iteriral čez ndarray-e in za vsak element ločeno izračunal vsote vrst.

3.1 Maclaurinova vrsta

Za majhne vrednosti argumenta x lahko Ariyjevi funkciji izrazimo z Maclaurinovima vrstama:

$$\mathrm{Ai}(x) = \alpha f(x) - \beta g(x)\,, \qquad \mathrm{Bi}(x) = \sqrt{3}\left[\alpha f(x) + \beta g(x)\right].$$

Dodatno v $x = 0$ veljata zvezi

$$\alpha = \mathrm{Ai}(0) = \mathrm{Bi}(0)/\sqrt{3} \approx 0.355028053887817239,$$
$$\beta = -\mathrm{Ai}'(0) = \mathrm{Bi}'(0)/\sqrt{3} \approx 0.258819403792806798.$$

Vrsti za $f(x)$ in $g(x)$ pa lahko zapišemo kot

$$f(x) = \sum_{k=0}^{\infty} \frac{\Gamma(\frac{1}{3}+k)}{\Gamma(\frac{1}{3})} \frac{3^k x^{3k}}{(3k)!}\,, \qquad g(x) = \sum_{k=0}^{\infty} \frac{\Gamma(\frac{2}{3}+k)}{\Gamma(\frac{2}{3})} \frac{3^k x^{3k+1}}{(3k+1)!}\,.$$

Člene vrst sem računal rekurzivno. *Rising factorial* se enostavno lahko zapiše rekurzivno z uporabo identitete za Gama funkcijo

$$\frac{\Gamma(z+n)}{\Gamma(z)} = z(z+1)\ldots(z+n-1).$$

Rekurzivno lahko zapišem še fakulteto in potenco števila 3. Tako sem dobil rekurzivni zvezi

$$f_{k+1}(x) = f_k(x) \frac{3x^3(\frac{1}{3}+k-1)}{(3k)(3k-1)(3k-2)}\,,$$
$$g_{k+1}(x) = g_k(x) \frac{3x^3(\frac{2}{3}+k-1)}{(3k+1)(3k)(3k-1)}\,.$$

3.2 Asimptotski vrsti

Uvedemo novo spremenljivko $\xi = \frac{2}{3}x^{3/2}$. Za velike positivne vrednosti argumenta x lahko zapišemo asimptotski razvoj

$$\mathrm{Ai}(x) \sim \frac{\mathrm{e}^{-\xi}}{2\sqrt{\pi}x^{1/4}} L(-\xi)\,, \qquad \mathrm{Bi}(x) \sim \frac{\mathrm{e}^{\xi}}{\sqrt{\pi}x^{1/4}} L(\xi)\,.$$

Za velike negativne vrednosti pa

$$\mathrm{Ai}(x) \sim \frac{1}{\sqrt{\pi}(-x)^{1/4}}\Big[\quad \sin(\xi - \pi/4)\, Q(\xi) + \cos(\xi - \pi/4)\, P(\xi)\Big]\,,$$
$$\mathrm{Bi}(x) \sim \frac{1}{\sqrt{\pi}(-x)^{1/4}}\Big[-\sin(\xi - \pi/4)\, P(\xi) + \cos(\xi - \pi/4)\, Q(\xi)\Big]\,.$$

Tu vrste $L(x)$, $P(x)$ in $Q(x)$ zapišemo kot

$$L(z) \sim \sum_{s=0}^{\infty} \frac{u_s}{z^s}\,, \qquad P(z) \sim \sum_{s=0}^{\infty} (-1)^s \frac{u_{2s}}{z^{2s}}\,, \qquad Q(z) \sim \sum_{s=0}^{\infty} (-1)^s \frac{u_{2s+1}}{z^{2s+1}}\,,$$

kjer so koeficienti u_s

$$u_s = \frac{\Gamma(3s + \frac{1}{2})}{54^s s!\, \Gamma(s + \frac{1}{2})}\,.$$

Tudi tu bi se dalo člene vrste računati rekurzivno z uporabo še ene identitete za gama funkcijo najdena na MathWorld [1], ki pravi

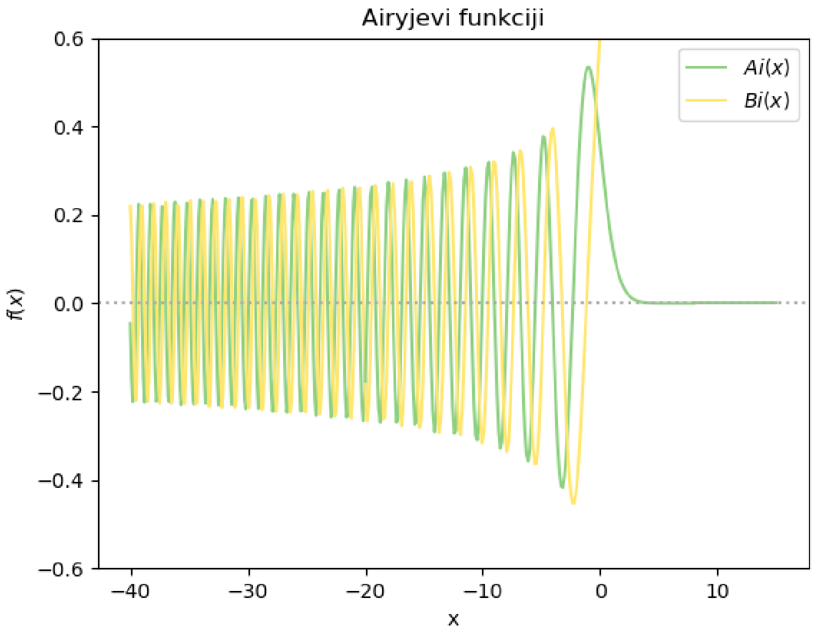
$$\Gamma(s + \frac{1}{2}) = \frac{\sqrt{\pi}(2s-1)!!}{2^s}; s \in \mathbb{N}\,.$$

Žal mi je nekoliko zmanjkalo časa, da bi to lahko uspel premisliti in pravilno zapisati.

V `mpmath` sem nastavil natančnost računanja na 100 decimalnih mest. Razlog za to argumentiram kasneje. Za izračun Maclaurinove vrste v programu določim maksimalno število iteracij oz. maksimalno število členov. Člene vrste sešteva dokler nadaljnji, pri določeni natančnosti, ne prispevajo več k rezultatu. Maksimalno število iteracij za Maclaurinovo vrsto sem nastavil na 1000. Običajno sešteje program okoli 150 členov. Podobno za izračun asimptotskih vrst. Določim maksimalno število iteracij in člene neham seštevati, ko začnejo naraščati. Maksimalno število iteracij za asimptotske vrste za pozitivne argumente sem nastavil na 50. V večini primerov je program seštel okol 30 členov. Maksimalno število iteracij asimptotske vrste za negativne argumente pa na 5, kjer je program običajno seštel 3 člene.
Kot referenčni funkciji sem uporabil Ariyjevi funkciji, ki sta vgrajeni v `mpmath`. Njuni natančnosti pa sem preveril z Airyjevi funkciji, ki sta vgrajeni v `SciPy`.

4 Rezultati

Približke lahko sestavimo skupaj, da dobimo takšno sliko kot na začetku z vgrajenimi funkcijami. V točki $x = -20$ sem zlepil skupaj asimptotsko vrsto za negativne argumente in Maclaurinovo vrsto, v točki $x = 8$ pa Maclaurinovo vrsto z asimptotsko vrsto za pozitivne argumente. Veliko število decimalk pri računanju mi je omogočilo, da je Maclaurinova vrsta zelo natančna tudi precej daleč v negativna števila.

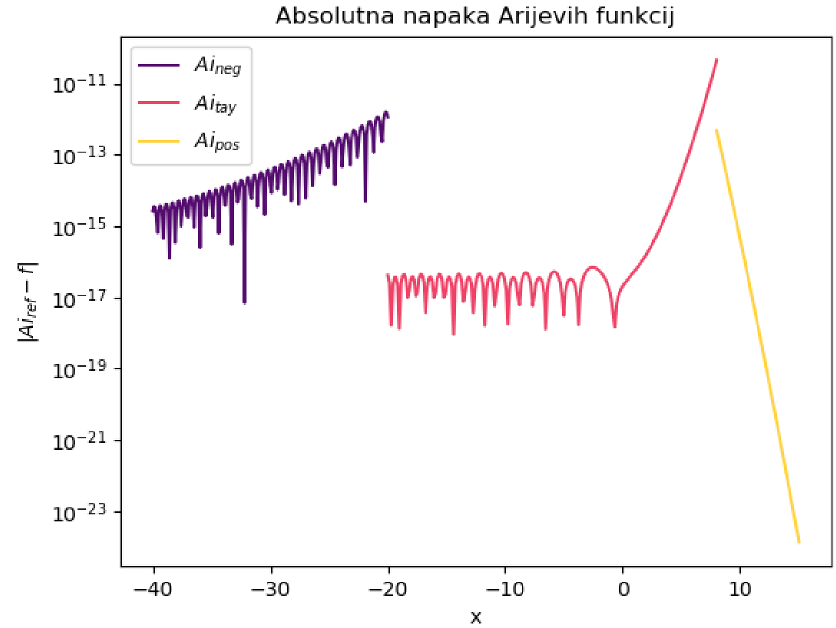


Slika 2: Graf zlepka približkov Ariyjevih funkcij za realne argumente

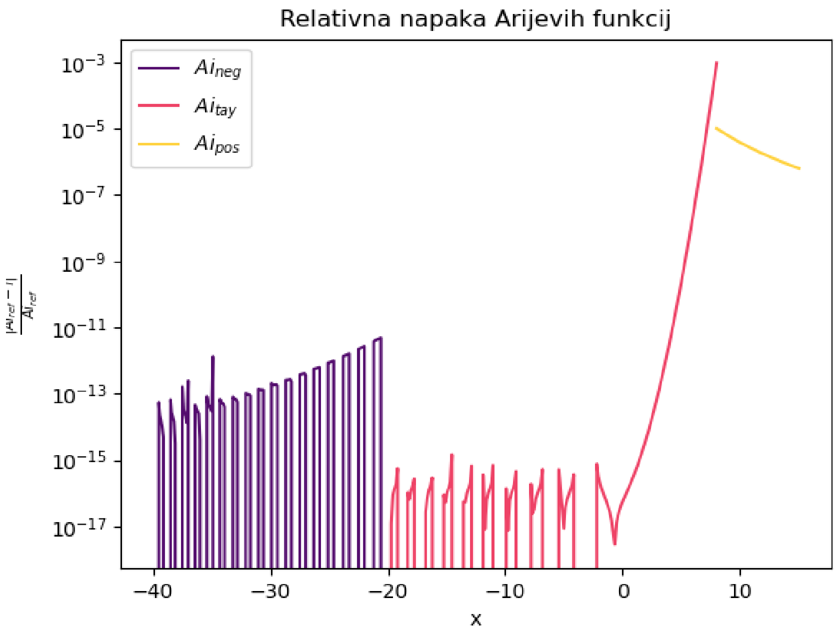
4.1 Funkcija Ai

Absolutno napako funkcije sem izračunal kot $|\mathrm{Ai}_{ref} - f|$, kjer f predstavlja vsote vrst. Na grafu sem jih obarval različno. Vidimo, da ni nobenih napak pri doseganju natančnosti 10^{-10}. Opazi se, da za $x > 3$ absolutna napaka Maclaurinove vrste hitro raste. Z natančnostjo asimptotske vrste pri $x < 8$ sem imel težave. To sem rešil z velikim številom decimalk in iteracij, ki mi je omogočilo Maclaurinovo vrsto uporabiti dokaj daleč od izhodišča.

Relativno napako sem izračunal kot $|\mathrm{Ai}_{ref} - f|/\mathrm{Ai}_{ref}$. Relativne napake mi nikakor in uspelo spraviti pod zahtevano natančnost za velike argumente. Sumim, da je razlog v tem, da se nam pravzaprav pojavi "$\frac{0}{0}$". Vrednost funkcije gre hitro proti nič in absolutna napaka je tudi majhna. Kvocient teh dveh malih števil pa je število, ki je očitno signifikantno.



Slika 3: Graf absolutne napake Ai

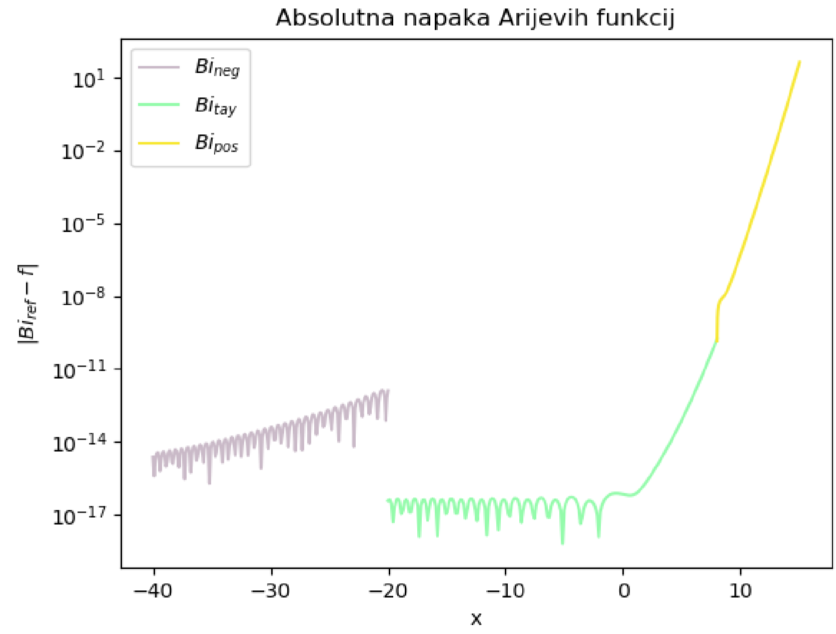


Slika 4: Graf relativne napake Ai

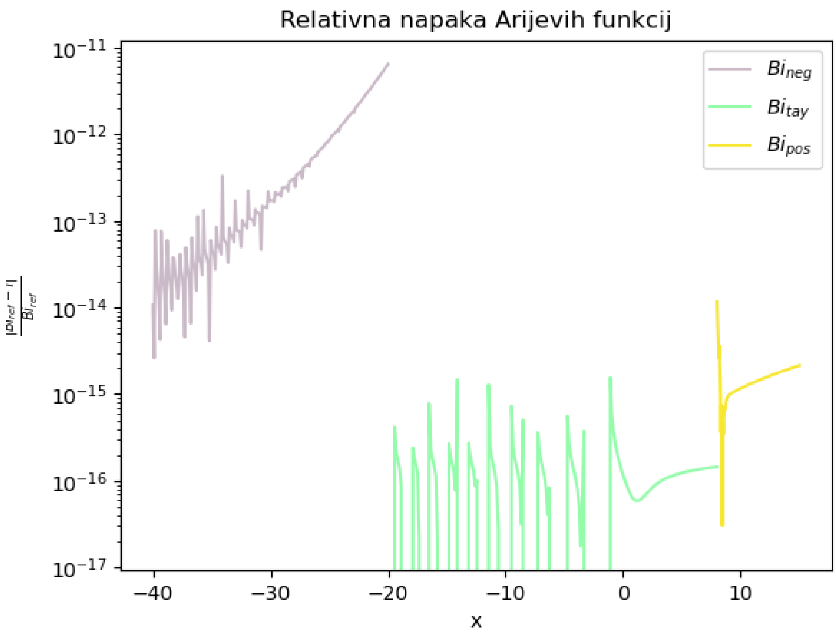
4.2 Funkcija Bi

Tako kot prej sem absolutno napako izračunal kot $|\mathrm{Bi}_{ref} - f|$, kjer f predstavlja vsote vrst. Opazimo zanimiv pojav. Absolutna napaka za velike vrednosti argumenta zelo hitro naraste. To sicer ni problem, ker tam tudi funkcija divje divergira in se napaka ne pozna za res, kar bomo lahko vidli na grafu relativne napake.

Izračunano tako kot prej, je relativna napaka $|\mathrm{Bi}_{ref} - f|/\mathrm{Bi}_{ref}$. Kot napovedano, je relativna napaka za velike x res majhna zaradi zelo hitrega naraščanja funkcije.



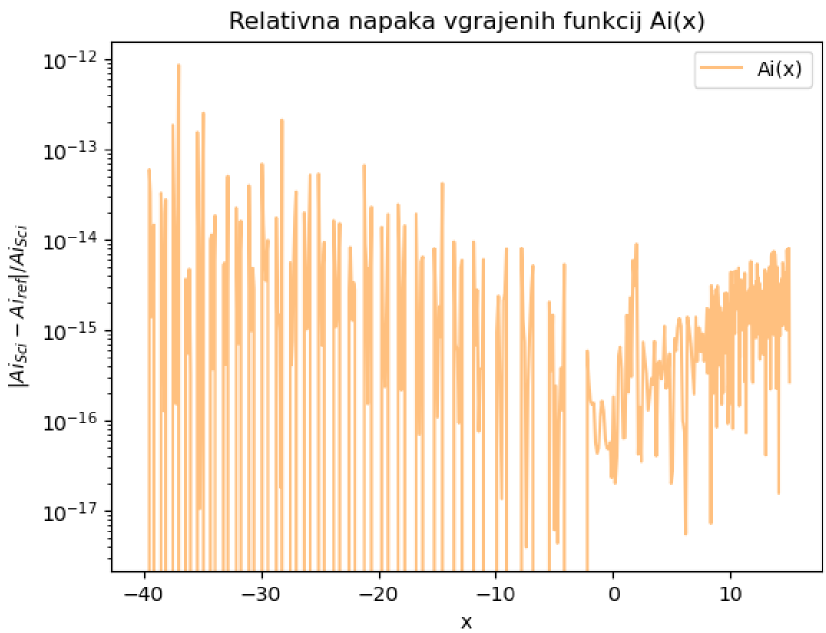
Slika 5: Graf absolutne napake Bi



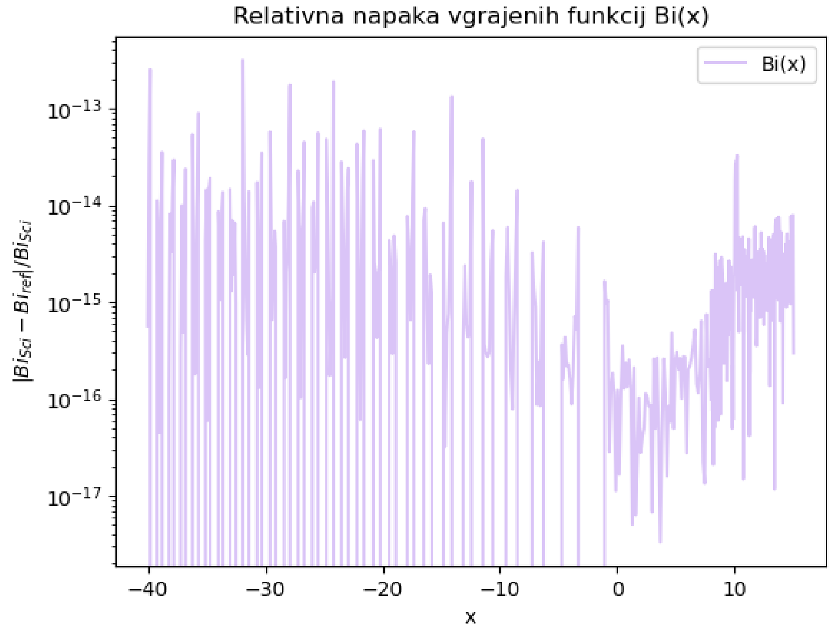
Slika 6: Graf relativne napake Bi

4.3 Natančnost `mpmath.airy`

Pomembno se je zavedati, da tudi naša referenčna funkcija ni perfektno natančna ampak je tudi nek približek. Da bi preveril, kako natančna je, sem uporabil še vgrajene Airyjeve funkcije iz paketa `SciPy` in sem narisal relativni napaki izračunani kot $|\mathrm{Ai}_{Sci} - \mathrm{Ai}_{ref}|/\mathrm{Ai}_{Sci}$ in $|\mathrm{Bi}_{Sci} - \mathrm{Bi}_{ref}|/\mathrm{Bi}_{Sci}$, kjer se indeks *ref* navezuje na referenčno funkcijo iz `mpmath` in indeks *Sci* na primerjalni funkciji iz `SciPy`. Seveda sta tudi funkciji iz `SciPy` približni. Primerjava se mi zdi smiselna, ker sta `mpmath` in `SciPy` neodvisna in je kot bi ju primerjal z neodvisno metodo. Vidimo, da je relativna napaka obeh Ariyjevih funkcij v primerjavi z `Scipy`-jevmi pod 10^{-12}. To je za naše zahteve natančnosti na manj kot 10^{-10} dovolj dobro. Pri večji željeni natančnosti pa bi lahko povzročalo probleme. Za večjo natančnost bi moral sam dovolj dobro numerično integrirati integralska zapisa za Ariyjeve funkcije.



Slika 7: Graf relativne napake referenčne funkcije Ai_{ref}



Slika 8: Graf relativne napake referenčne funkcije Bi_{ref}

4.4 Komentarji in možne izboljšave

Zdi se mi, da sem večino pomembnih ugotovitev že sproti komentiral. Najbolj očitna izboljšava pri mojem programu je, da bi zmanjšal število decimalnih mest s katerimi računam. Uspešno jih lahko znižal na 30, ampak sem počasi začel imeti težave z natančnostjo Maclaurinove vrste. Definitivno natančnost na 100 decimalk znatno prispeva k učinkovitosti in "run time-u programa. Vsekakor pa bi tudi koristilo premisliti in narediti rekurizivne zapise še za obe asimptotski vrsti.

Literatura

[1] Eric W. Weisstein. Gamma function from wolfram mathworld, Sep 2021. Dostopno na `http://mathworld.wolfram.com/GammaFunction.html`; Zadnje obiskano 11.10.2021.