4. Domača naloga - Grafi II

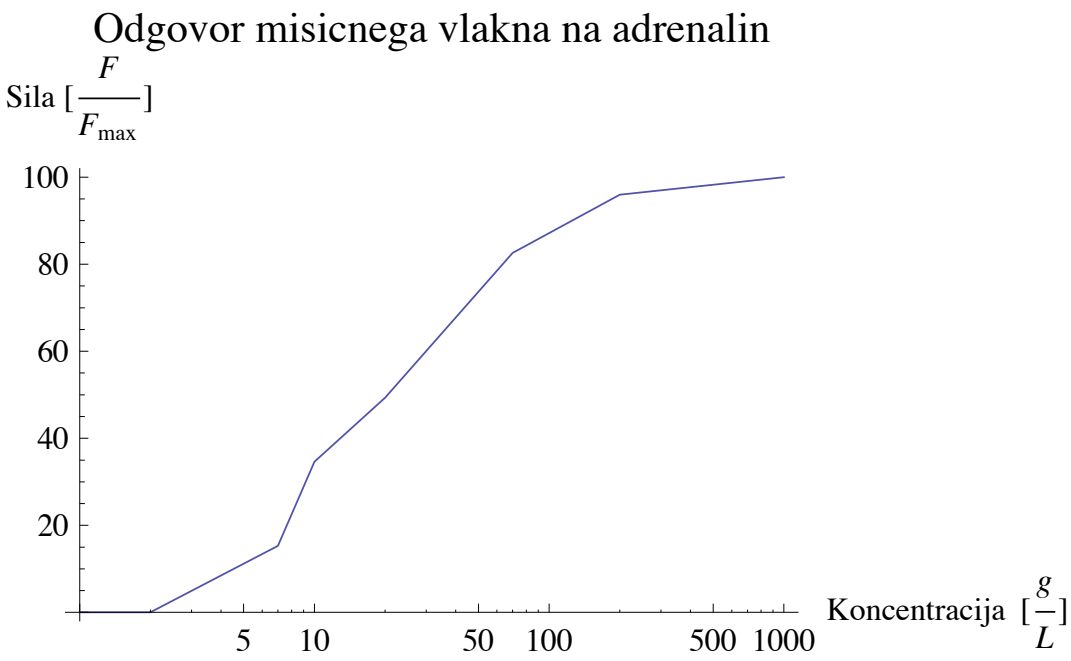
Urban Škudnik

2009-04-02

1 Adrenalin

Opis V prvem delu naše vaje smo se osredotočili na graf, ki smo ga v preteklosti že videli - to je graf, pri katerem smo mišično vlakno iz žabjega srca oblivali z adrenalinom, pri čemer smo merili skrčevanje mišice ter podatke normirali pri največji koncentraciji adrenalina.

Tokrat smo namesto običajnega (linearnega) prikaza grafa na abscisni osi raje uporabili logaritemsko skalo, kar nam veliko bolje prikaže spreminjanje raztezka mišice.

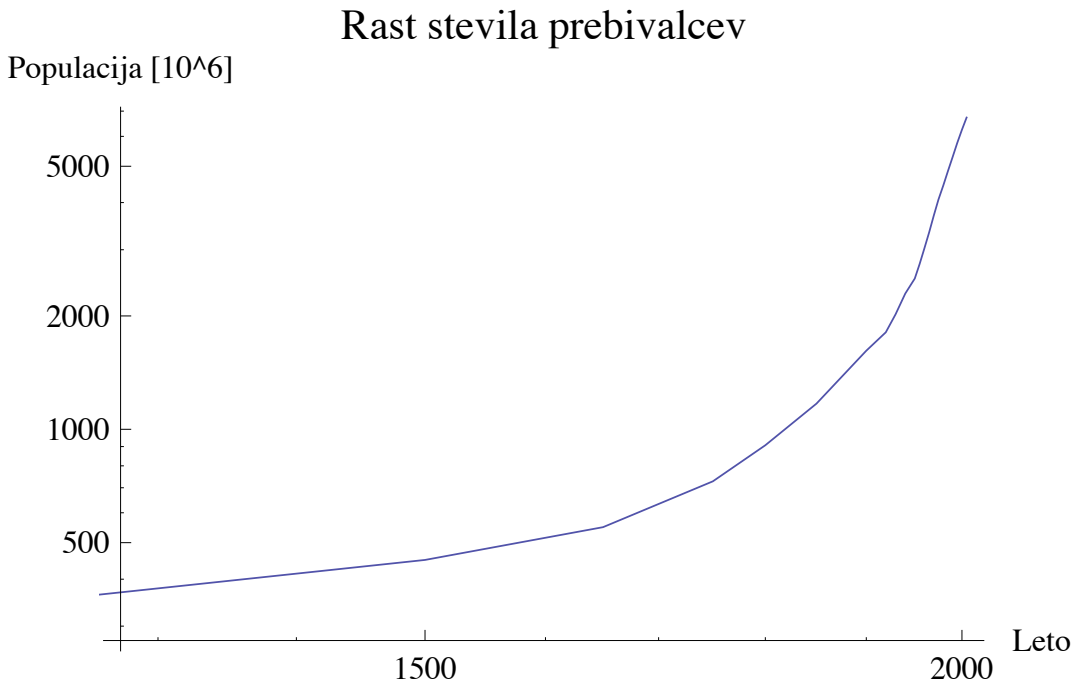


Sigmoidni graf

Tehnični opis Graf je bil narisan v programu *Mathematica* s pomočjo ukaza `ListLogLinearPlot[data, Joined -> True]` .

2 Zgodovina človeštva

Opis V drugem delu naše naloge smo morali prikazati rast človeške populacije skozi zgodovino. Ker se je slednja močno povečala tekom zadnjega tisočletja se je za najboljši graf izkazal graf, ki je imel tako na abscisni kot tudi ordinatni osi logaritemsko skalo.



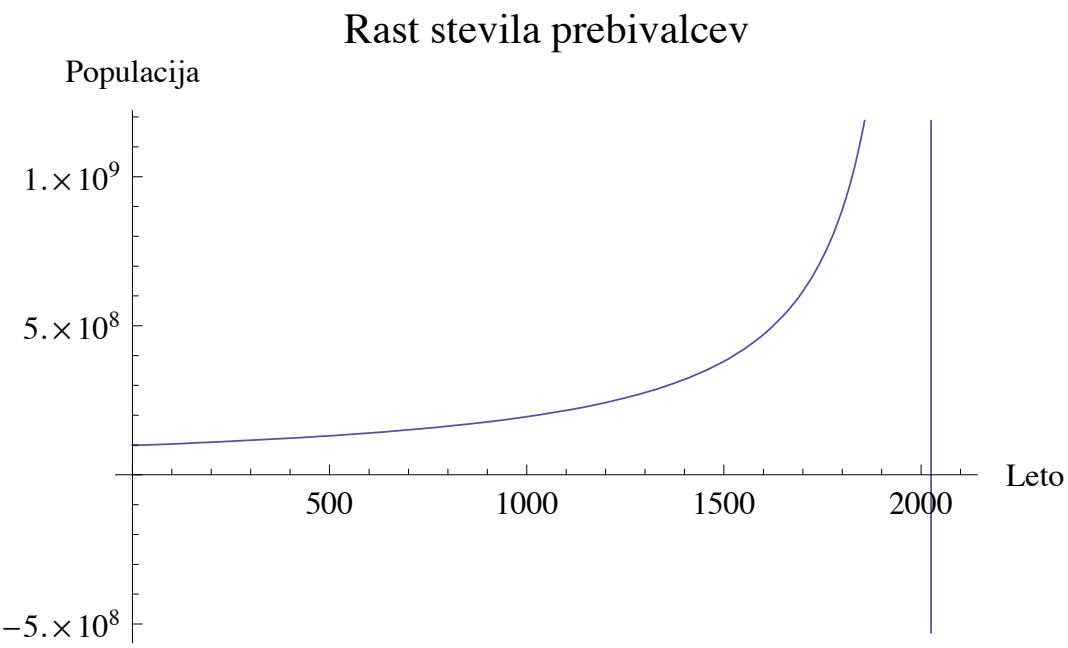
LogLog skala

Tehnični opis Za prikaz podatkov za realnost rast človeške populacije je zadoščala uporaba funkcije `ListLogLogPlot`, ki nam podatke iz seznama nariše na graf z logaritemsko skalo na abscisni in ordinatni osi.

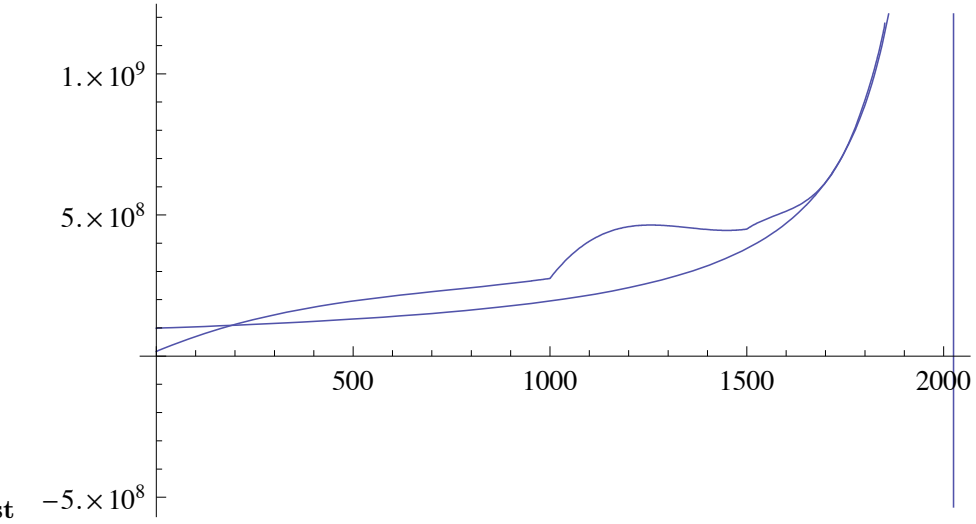
Opomba *Podatki na ordinatni osi so v milijonih.*

Funkcijski opis Ker je rast človeške populacije lahko opisljiva s funkcijo $\frac{2 \cdot 10^{11}}{2025-t}$ si lahko pogledamo tudi kako se ta funkcija ujema z interpelirano funkcijo za realno rast človeške populacije.

Funkcijska rast



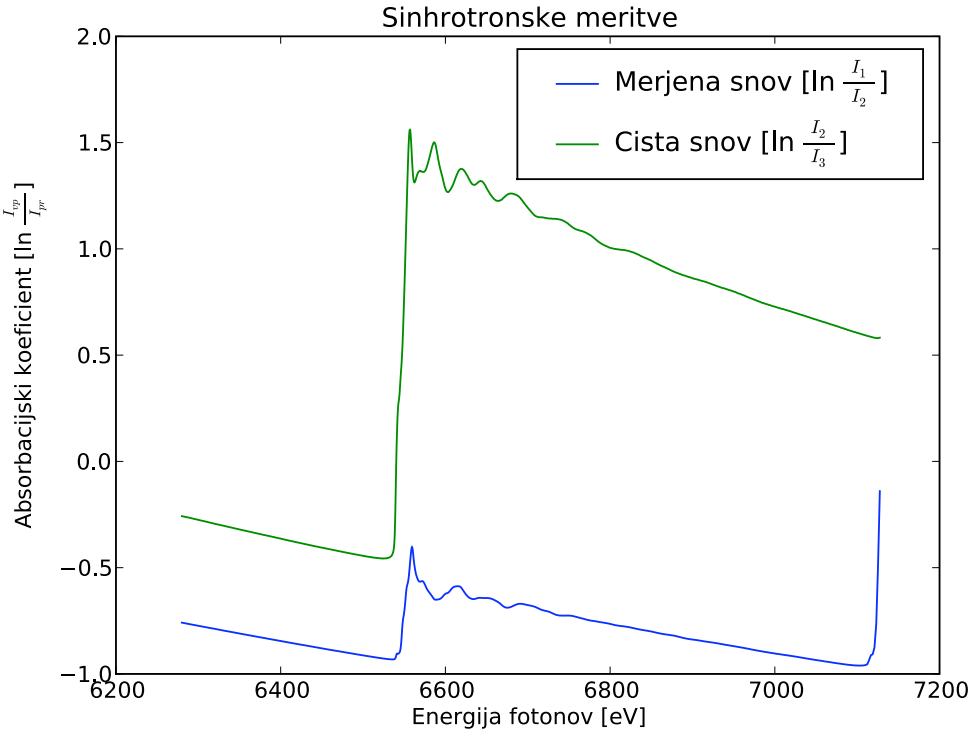
Interpolirana rast



Tehnični opis Za primerjavo s funkcijsko rastjo smo morali najprej v programu *Mathematica* uporabiti funkcijo `Interpolation`, ki nam iz danih podatkov poskuša izdelati najboljšo možno funkcijo, ki bi opisala spreminjanje podatkov. Ko vnesemo te podatke na graf vidimo, da se interpolacijska funkcija, kljub določenim popačenjem, močno ujema s funkcijo $\frac{2 \cdot 10^{11}}{2025-t}$, posebej v za nas najbolj zanimivem delu - ko človeška populacija naraste v zadnjem stoletju.

3 Sinhrotronske meritve

Opis Pri sinhrotronskih meritvah smo morali prikazati razliko v absorbacijskem spektru, če vstavimo v sinhrotron nek merjenec ali pa če vstavimo v njega čisto snov, s katero primerjamo.



Graf sin. meritev

Tehnični opis Za izdelavo tega grafa nisem več uporabil *Mathematice*, pač pa sem posegel po orodju *Matplotlib* [http://matplotlib.sourceforge.net/], ki je knjižica za Python za risanje grafov. Na naslednji strani prilagam kodo, pri kateri s CSV modulom uvozim podatke, jih spremenim v seznam in iz njih izdelam podatkovne nize, ki se uporabijo za izris grafa.

Koda - Sinhrotronske meritve

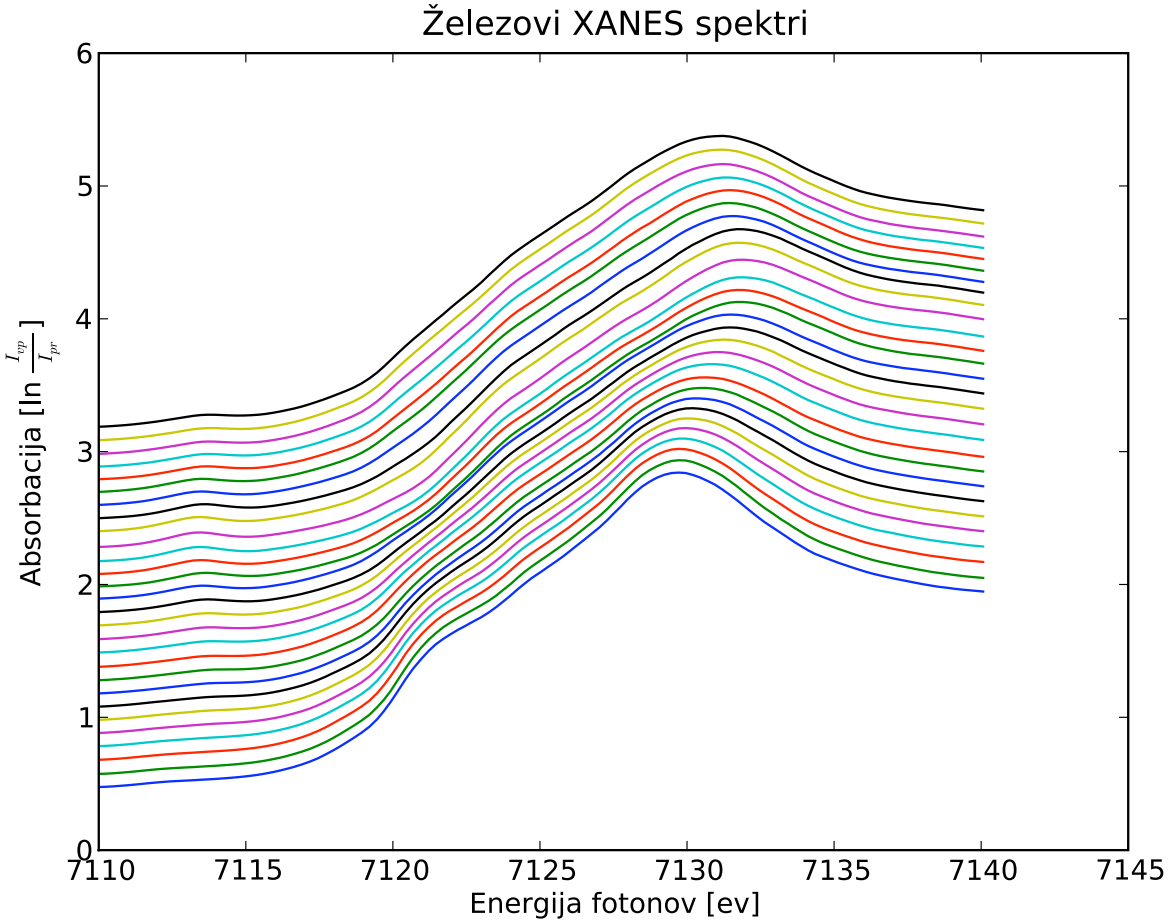
```
absorb = csv.reader(open("Md29mn_00001.fio.txt"), delimiter=" ")
rows = list()
for row in absorb:
    c = list()
    for column in row:
        if column:
            c.append(column)
    rows.append(c)

set1 = [(x[0], math.log(float(x[5])/float(x[6]))) for x in rows]
set2 = [(x[0], math.log(float(x[6])/float(x[7]))) for x in rows]

plot.plot([e for e, l in set1], [l for e, l in set1], label=r"Merjena snov")
plot.plot([e for e, l in set2], [l for e, l in set2], label=r"Cista snov")
plot.xlabel("Energija fotonov [eV]")
plot.ylabel(r"Absorbacijski koeficient")
plot.title("Sinhrotronske meritve")
plot.legend()
plot.show()
```

4 XANES

Opis Na koncu smo merili še absorbacijske spektre robove železa v novi litijevi ionski bateriji med njenim polnjenjem in praznjenjem, pri čemer smo zapisali signal XANES oz. absorbacijo v odvisnosti od energije fotonov. Vsaka črta je bila merjena dvajset minut za njeno predhodnico.



XANES

Tehnični opis Podobno kot pri sinhrotronu sem tudi tukaj posegel po orodju *Matplotlib*, s katerim sem po začetni inicializaciji podatkov uporabil relativno preprosto rešitev za izris 28 meritev.

Vsaka meritev spektra je tudi dvignjena nad predhodnico za dodatno tridesetino vrednosti v primerjavi z predhodnjo meritvijo, kar nam omogoča lažje opazovanje razlik v spektrih.

Koda

```
absorb = csv.reader(open("Fe_rob_0_27.xmu.txt"), delimiter=" ")
rows = list()
for row in absorb:
    c = list()
    for column in row:
        if column:
            c.append(column)
    rows.append(c)


xset = list()
for row in rows:
    xset.append(row[0])

def get_column_data(data, column):
    c = list()
    for row in data:
        c.append(row[column])
    return c

for x in range(1, 29):
    plot.plot(xset, [float(i)+(float(x)/10) for i in get_column_data(rows, x)])

plot.xlabel("Energija fotonov [ev]")
plot.ylabel(r"Absorbacija [ln $\frac{I_{iz}}{I_{vp}}$]")
plot.title(u"Železovi XANES spektri")
plot.show()
```