

University of *Ljubljana*
Faculty of *Mathematics and Physics*



Modelska Analiza 1

3. naloga - Numerična minimizacija

Avtor: Matic Lubej
Asistent: dr. Simon Čopar
Predavatelj: prof. dr. Alojz Kodre

Ljubljana, 22.10.2013

Naloga:

Pri tem poročilu smo se seznanili z numerično minimizacijo, ki nam omogoča tudi reševanje nelinearnih optimizacijskih problemov, kar z metodami iz prejšnjega poročila ni bilo mogoče. V prvem delu naloge smo imeli t.i. Thomsonov problem, kjer smo na prevodno kroglo nanegli enake klasične naboje in poiskali stanje z najnižjo elektrostatično energijo. V drugem delu naloge smo spet reševali problem optimalne vožnje, kjer smo tokrat dali poudarek na numerično minimizacijo, ki nam je omogočila dodajanje poljubnih „elastičnih vezi“, ki smo jih nastavili z eksponentno funkcijo. V obeh delih naloge sem računal in grafično predstavil z matematičnim orodjem Mathematica.

Del I

Thomsonov problem

Na prevodno kroglo nanesemo N enakih (klasičnih) nabojev. Kako se razmestijo po površini? Zahtevamo seveda minimum elektrostatične energije. Uporabi katero od minimizacijskih metod, npr. Powellovo ali n -dimenzionalni simpleks (amebo).

1 Teorija

Za elektrone privzamemo, da so točkasti delci, zato da jim lahko pripišemo električno polje nabite točke:

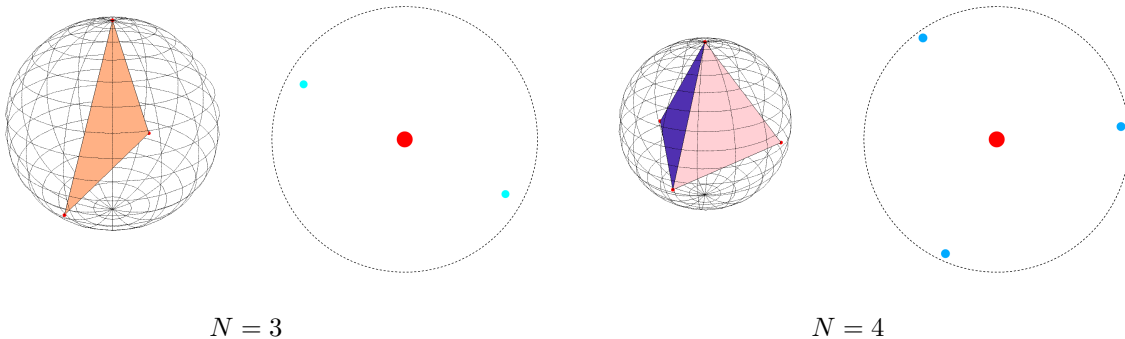
$$V(r) = k_e \frac{q_1 q_2}{r}, \quad (1)$$

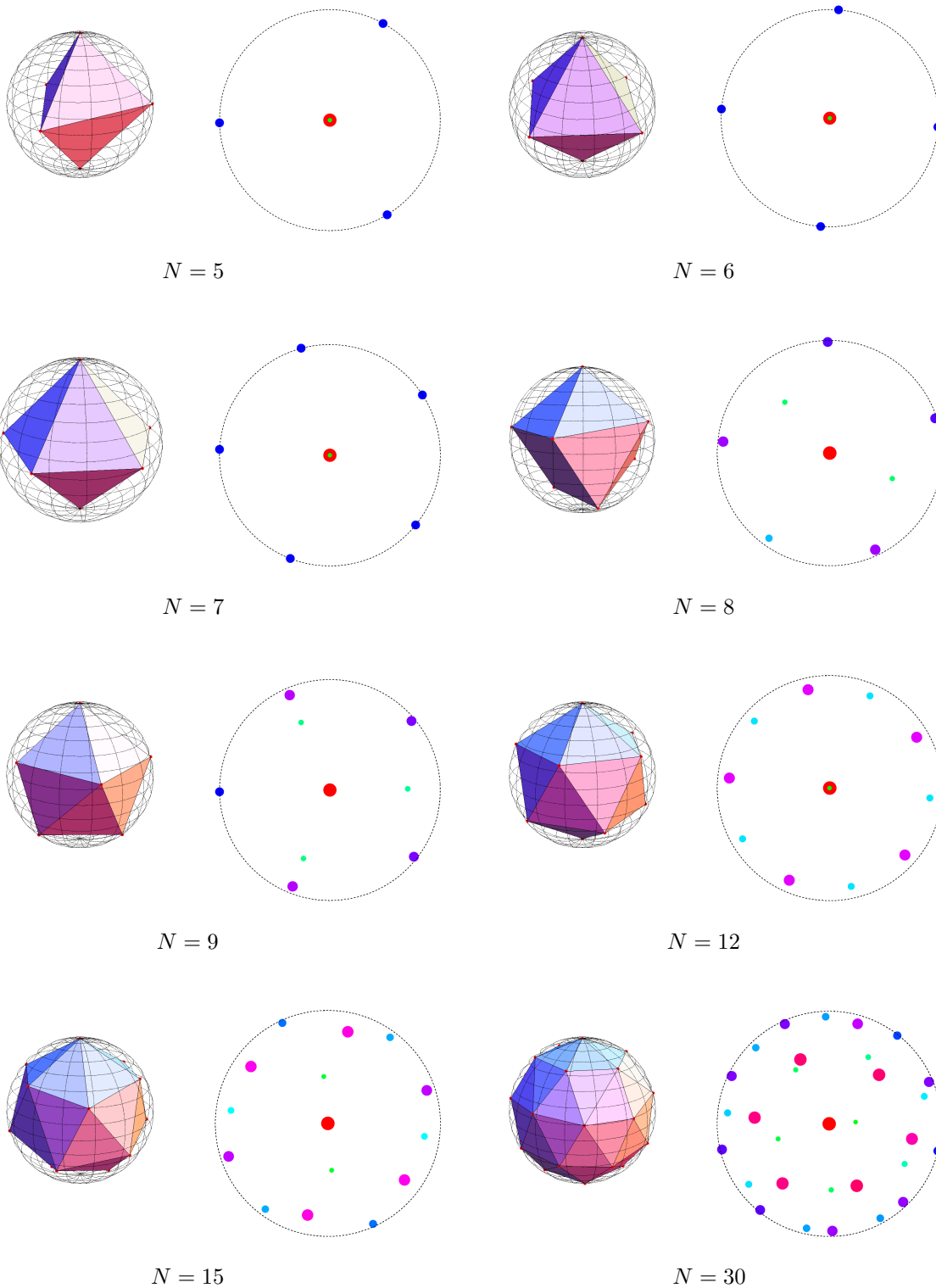
kjer je konstanta $k_e = \frac{1}{4\pi\epsilon_0}$ in jo bomo postavili na $k_e = 1$ brez izgube splošnosti, q_1 in q_2 pa sta naboj prvega in drugega delca. Ker imamo opravka le z elektroni, bo ta izraz vedno enak $q_1 q_2 = e_0^2 = 1$, kjer smo vrednost osnovnega naboja spet brez izgube splošnosti postavili na enoto. V primeru N elektronov moramo sešteti skupno elektrostatično energijo, ki je vsota vseh parskih elektrostatičnih prispevkov:

$$V = \sum_{i \neq j} V_{ij} = \sum_{i \neq j} \frac{1}{\|\mathbf{r}_i - \mathbf{r}_j\|}.$$

2 Reševanje

Namesto kartezičnih koordinat sem se raje odločil za sferične, ker mora oddaljenost elektronov od središča krogle biti venomer enaka. S tem sem prihranil nekaj računanja, saj imam tako le dve prostostni stopnji na elektron. Iskal sem torej optimalni koordinati $\{\phi_i, \theta_i\}$, kjer sem jih v primeru prvega elektrona obe postavil na 0. Uporabil sem minimizacijsko funkcijo `NMinimize`, ki za argument sprejme količino, ki jo mora minimizirati in pogoje, ki jih mora upoštevati. Rešitve sem nato prikazal na dva načina; kot točke na krogli, ki so povezane med seboj z lupino in kot diski v krogu, kjer barva in velikost diska ustrežata višini elektrona (velika rdeča pika je elektron, ki je najvišje). Oglejmo si te rešitve za nekaj osnovnih vrednosti N :

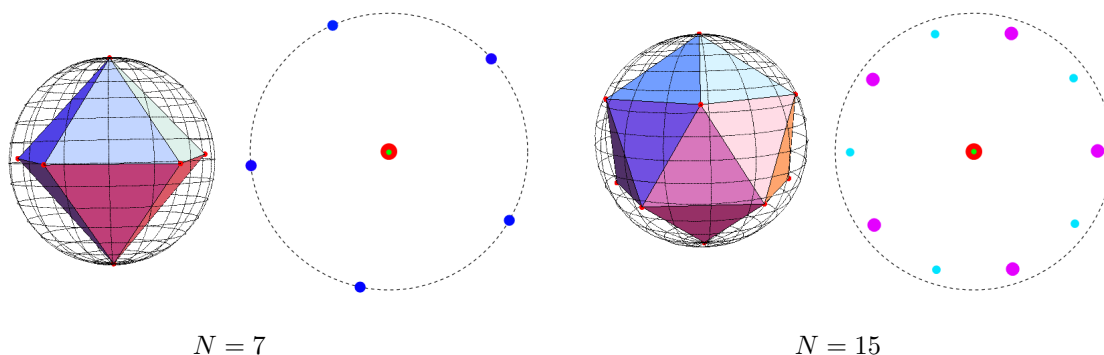




Vidimo, da če je število elektronov N ravno pravšnje, dobimo zaradi simetrije telesa, ki jih dobro poznamo pod imenom poliedri. Za lažjo predstavo teles prilagam še animacijo, ki je dostopna na povezavi: <https://dl.dropboxusercontent.com/u/12747812/Animacije/multi.gif>.

3 Izmišljena metoda

Za začetek sem se odločil, da metodo poizkusim narediti sam. Odločil sem se za najbolj enostavno verzijo, torej naključni skoki. Na začetku sem naključno izbral vektor parametrov ϕ_i in θ_i nato pa v vsakem naslednjem koraku spremenil enega od parametrov in preveril, če ima izbrana konfiguracija nižjo elektrostatično energijo. Če je bila energija nižja, sem spremembo sprejel, drugače zavrgel. Spremembe sem izvajal na dva načina; prvi način so bili čisto naključni skoki parametrov ϕ_i in θ_i na območjih $[0, 2\pi]$ in $[0, \pi]$, drugi način pa so bili manjši koraki okoli trenutnih vrednosti parametrov. S prvim načinom so bile optimalne vrednosti izbrane na grobo, z drugim pa bolj natančno. Oglejmo si rezultate te metode:



Na prvi pogled razik ni videti, vendar je bila časovna potrata te metode občutno večja, kar pa ni presenetljivo, saj imajo prave metode mnogo bolj kompleksne mehanizme, ki v najkrajših možnih časih najdejo optimalne konfiguracije. Prilagam še animacijo za števili elektronov $N = 5$ in $N = 12$, ki prikazujeta posamezne korake pri iskanju optimalne konfiguracije za izmišljeno metodo:

https://dl.dropboxusercontent.com/u/12747812/Animacije/random_n5.gif,

https://dl.dropboxusercontent.com/u/12747812/Animacije/random_n12.gif.

4 Napredne metode

Mahematica in minimizacijska metoda NMinimize mi omogočata izbiro različnih metod minimizacije, s funkcijo StepMonitor pa mi omogočata celo vpogled v posamezne korake celotne iteracije. Ker so rezultati vseh metod enaki, je ta razdelek bolj besedilen. Vse prejšnje grafe sem risal brez definirane metode, kar pomeni, da je metoda bila izbrana avtomatsko tako, da je reševanje najbolj optimalno. V vseh primerih se je izkazalo, da imajo metode trenutke mirovanja. Odločil sem se, da ta mirovanja odstranim, saj ne predstavljajo nič pomembnega, tako da so prikazane le spremembe. Oglejmo si delovanje teh osnovnih metod, kjer sem zraven priložil še animacije postopkov.

4.1 Metoda „Nelder-Mead“

Ta metoda je znana tudi pod imenom n-dimenzionalni simplex, oz. „ameba“ je metoda, ki deluje dobro za funkcije, ki nimajo veliko lokalnih minimumov. Izkazala se je za najhitrešo metodo pri tem problemu. Postopek delovanja te metode:

<https://dl.dropboxusercontent.com/u/12747812/Animacije/NM.gif>.

Vidimo, da metoda hitro najde grobe vrednosti točk, ki jih nato počasi popravi.

4.2 Metoda segrevanja („Simulated Annealing“)

To metodo si lahko predstavljamo kot segrevanje kovine, kjer nato dovolimo da se počasi ohladi in doseže minimum. V vsakem koraku iteracije generiramo novo konfiguracijo, ki jo sprejmemo, če je pogoj izpolnjen, drugače pa zavržemo z verjetnostjo $p = e^{-\frac{\Delta f}{10} \log(i+1)}$, kjer je i trenutna iteracija, Δf pa sprememba

vrednosti količine, ki jo hočemo minimizirati. Ta metoda preizkusi več začetnih pozicij točk in skuša najti optimalno začetno pozicijo za vsako posebej. Postopek delovanja te metode:

<https://dl.dropboxusercontent.com/u/12747812/Animacije/SA.gif>.

Vidimo, da metoda večkrat preizkusi različne začetne pozicije elektronov, dokler ne najde optimalnih.

4.3 Metoda diferencialnega razvoja („Differential Evolution“)

Kot ime nakazuje, ta metoda deluje na principu razlik. Metoda izbere nove točke na kompleksen način in to ponavlja, dokler ne skonvergira v smislu, da je razlika med vrednostjo nove in stare količine, ter oddaljenost med novimi in starimi točkami manjša od tolerantne. Postopek delovanja te metode:

<https://dl.dropboxusercontent.com/u/12747812/Animacije/DE.gif>.

4.4 Metoda naključnega iskanja („Random Search“)

Približno delovanje te metode smo že spoznali, vendar je ta nekoliko bolj sofisticirana. Metoda generira naključne začetne točke, potem pa uporabi metodo lokalne optimizacije za vsako od začetnih točk. To stori večkrat, rešitev pa je konfiguracija z najmanjšo vrednostjo količine, ki jo minimizira. Postopek delovanja te metode:

<https://dl.dropboxusercontent.com/u/12747812/Animacije/RS.gif>.

Pri tej metodi vidimo, da je večino časa v bližini optimalne konfiguracije, kar je nekoliko neskladno z obnašanjem, ki ga bomo videli pozneje. Na prvi pogled je videti, da je ta metoda še najboljša, vendar se izkaže, da je časovno zelo potratna.

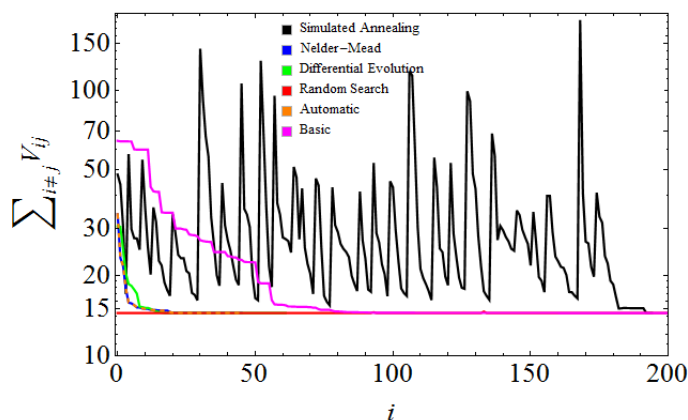
4.5 Avtomatsko izbrana metoda

Preveril sem še korake za avtomatsko izbrano metodo. Izkaže se, kot bomo videli tudi kasneje, da je način delovanja zelo podoben delovanju metode „Nelder-Mead“. Po tem lahko sklepamo, da je večinoma vsaj za naš problem bila ta metoda primarna izbira funkcije NMinimize. Postopek delovanja te metode:

<https://dl.dropboxusercontent.com/u/12747812/Animacije/AU.gif>.

5 Elektrostatična energija

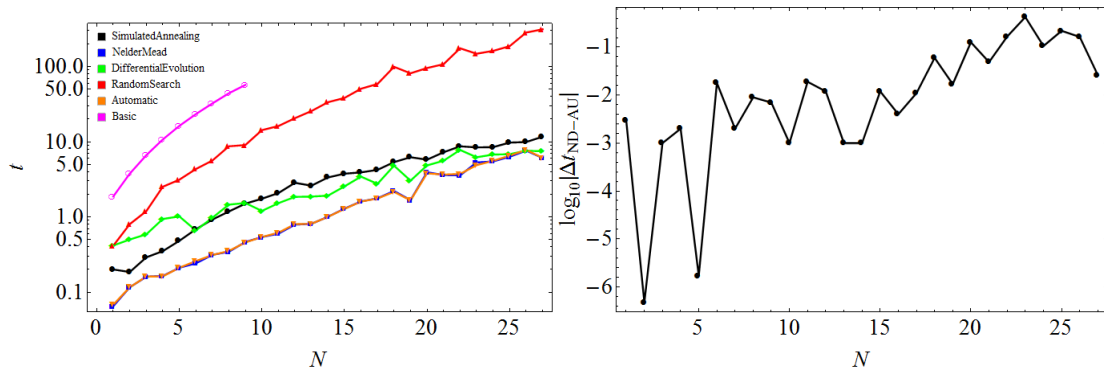
Odločil sem se še preveriti vrednosti elektrostatične energije v odvisnosti od števila iteracij pri posamezni metodi. Sklepamo lahko, da bo spreminjanje te količine sovpadalo z optimizacijo lokacij elektronov. Tako kot v prejšnjem razdelku, sem tudi tu prikazal le tiste iteracije, ki so spremenile konfiguracijo elektronov. Oglejmo si iskanje minimuma elektrostatične energije za različne metode, vključno s tisto, ki sem jo napisal sam („Basic“):



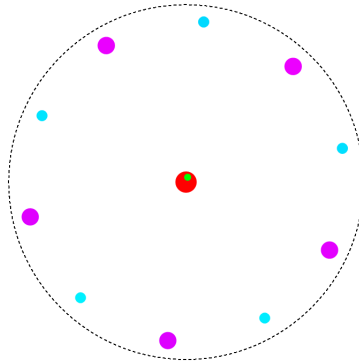
Kot smo pravilno sklepali, bi se res dalo določiti metodo po obliki padanja vrednosti elektrostatične energije. Vidimo, da se avtomatsko izbrana metoda in metoda „Nelder-Mead“ prekrivajo, torej funkcija NMinimize res deluje na ta način. Naivno bi sklepali, da je metoda „Random Search“ najbolj optimalna, vendar se v resnici izkaže, da zelo dolgo vztraja v tej poziciji. Metoda diferencialnega razvoja je podobne kakovosti kot „Nelder-Mead“, medtem ko metoda segrevanja in izmišljena metoda veljata za najmanj optimalni. Metoda segrevanja ima zelo divje delovanje, poleg tega pa še dolgo traja, medtem ko se je izmišljena metoda izkazala za povprečno, saj tudi da doseže optimum s počasnejšo minimizacijo in daljšim časom.

6 Časovna potratnost

Kot zanimivost sem se odločil preveriti še časovno potratnost funkcije NMinimize v odvisnosti od števila elektronov N , kar pa sem preveril za različne metode. Rezultati ne segajo daleč stran od pričakovanj, ki smo si jih o metodah ustvarili skozi poročilo. Oglejmo si rezultate na naslednjem grafu:



Ponovno lahko vidimo veliko podobnost med metodo „Nelder-Mead“ in avtomatsko izbrano metodo, ki sta tudi najhitrejši. Njuna razlika je na decimalnem nivoju, večja pa se z večjo vrednostjo števila N . Časovna potratnost metode diferencialnega razvoja je pričakovana, saj je le nekoliko slabša od metode „Nelder-Mead“. Nepričakovana pa sta rezultata za metodo segrevanja in metodo naključnega iskanja, saj bi po prejšnjih rezultatih rekli, da je metoda segrevanja najbolj potratna, vendar temu očitno ni tako. Vidimo lahko še časovno potrato izmišljene metode, ki je res najmanj optimalna. Časovna potrata pri tej metodi je odvisna od števila iteracij, kjer sem jih vzel toliko, da je zadnja vrednost pri tej metodi $N = 12$ še nekako natančna, kot lahko vidimo na naslednji sliki:



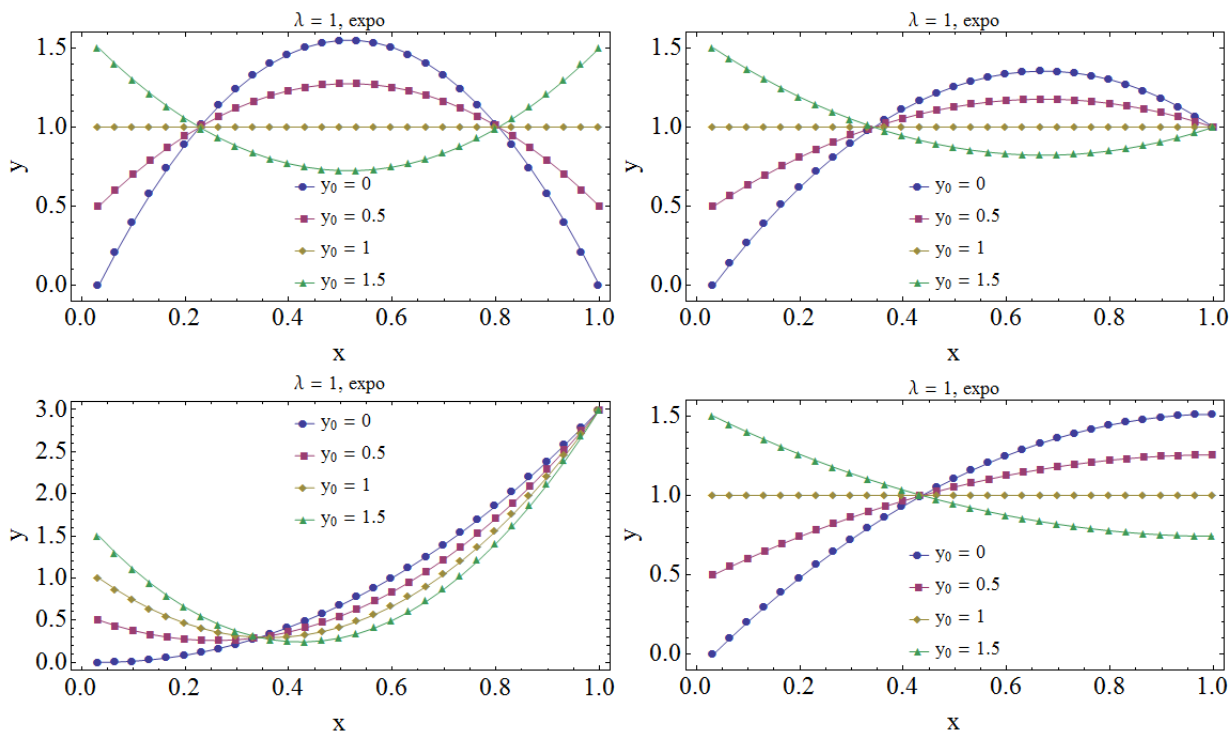
Del II

Problem optimalne vožnje - Numerična minimizacija

Problem optimalne vožnje skozi semafor, ki smo ga spoznali pri nalogi 1, lahko rešujemo tudi z numerično minimizacijo, če časovno skalo diskretiziramo. V prejšnjem poročilu sem že naredil podoben primer, kjer sem takšnemu reševanju dodal stroge omejitvene pogoje. V tem delu poročila pa smo lagrangianu $\mathcal{L} = \int \left(\frac{dy}{dt}\right)^2 dt - \lambda_{mul} \int y dt$ dodali še omejitve hitrosti v obliki členov

$$\mathcal{L}_{v_{max}} = e^{\lambda(y-y_{max})}, \quad (2)$$

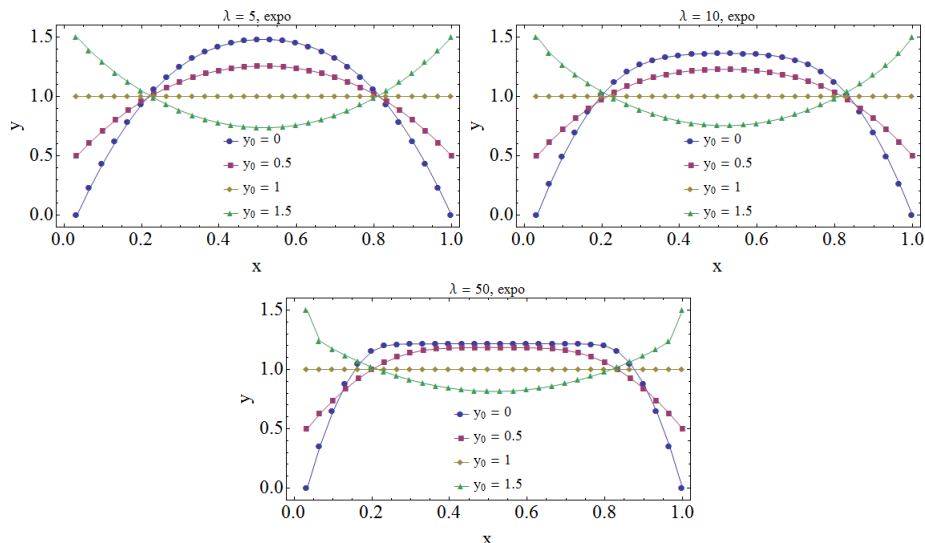
če hočemo približno zagotoviti pogoj $y \leq y_{max}$. Izpolnitev pogoja je toliko ostrejša, kolikor večjo vrednost λ vzamemo. Tudi tega problema sem se lotil z matematičnim orodjem Mathematica, kjer je bila spet najbolj uporabljena funkcija NMinimize. Omejitve sem preizkusil na treh različnih načinih vožnje za tri različne omejitvene funkcije, pri vsaki pa sem si ogledal vpliv različnih vrednosti parametra λ . Izkazalo se je, da je v vseh primerih rešitev z $\lambda = 1$ enaka rešitvam v prejšnjem poročilu, te pa so bile numerično reproducirane rešitve iz prvega poročila. Oglejmo si te osnovne rešitve za primer periodičnega, dveh fiksnih in dinamičnega robnega pogoja:



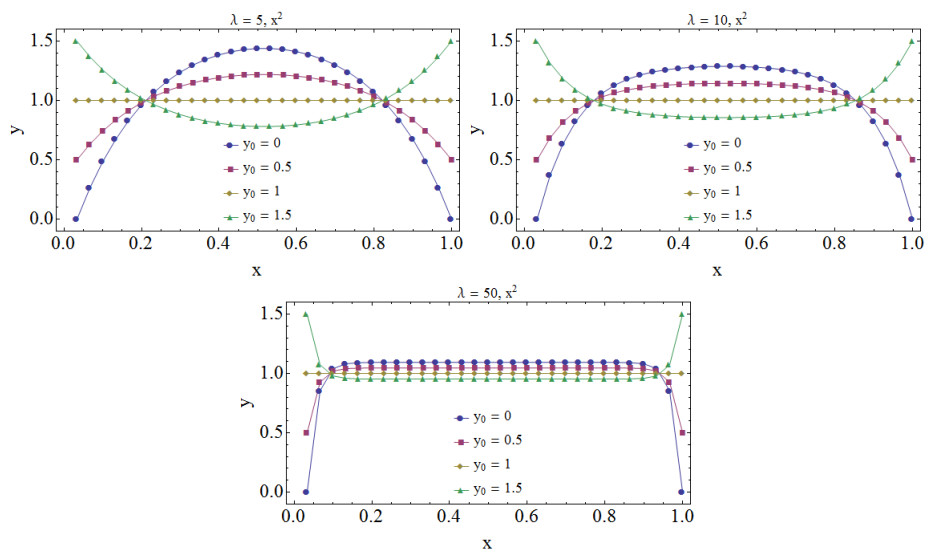
Vidimo, da numerične rešitve za vrednos parametra $\lambda = 1$ res sovpadajo z analitičnimi iz prvega poročila. Oglejmo si sedaj vsak način vožnje podrobneje.

1 Periodični robni pogoji

S pogojem $v_N = v_0$ dobimo periodične robne pogoje. V Lagrangian sem vključil eksponentni člen, kateremu sem spreminjal moč. Kot vemo, ima funkcija e^x na območju $x \leq 0$ vrednosti ≤ 1 , kjer eksponentno konvergira proti vrednosti 0, za $x \geq 0$ pa funkcija divergira. S takšno funkcijo omejimo hitrosti navzgor, kjer sem postavil omejitev $y_{max} = 1.2$, strogost omejitve pa spreminja vrednost λ :



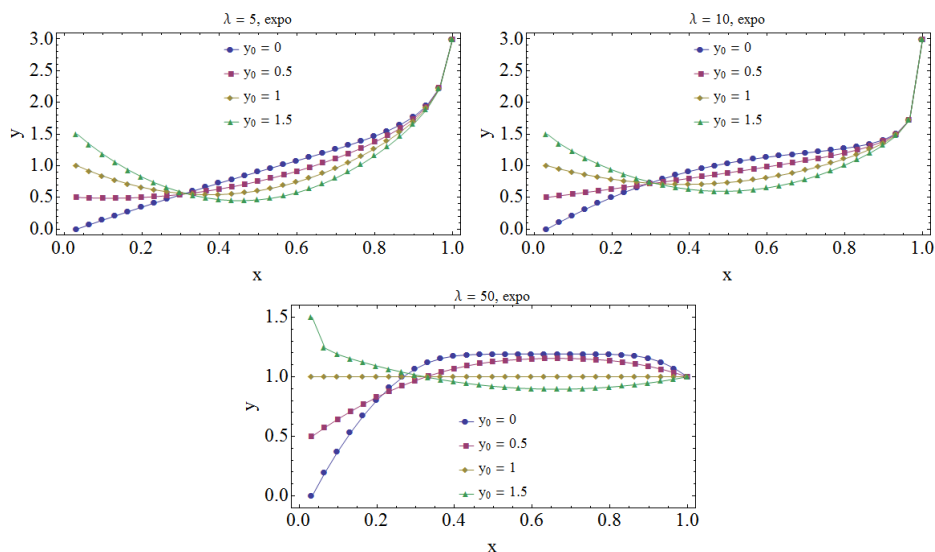
Vidimo, da z večanjem λ res vse bolj omejujemo hitrosti navzgor, pri čemer hitrosti, ki so pod omejitvijo, ostanejo praktično nespremenjene. Oglejmo si lahko še primer, kjer omejimo tudi hitrosti navzdol. Takšen primer bi lahko bil realističen recimo na avtocesti, kjer ne smemo voziti s hitrostjo, ki je manjša od minimalne. To lahko dosežemo s sodo funkcijo $e^{|x|}$, ali pa bolj priročno in enostavnejšo funkcijo x^2 . Ponovno si oglejmo rezultate:



V tem primeru je omejitvena funkcija vplivala tudi na hitrosti pod omejitvijo. Vidimo, da z večanjem λ manjšamo odmik hitrosti od v_{max} , kar je nekoliko nepriročno, saj hitrosti, ki so pod omejitvijo, večja proti vrednosti mejne hitrosti. Bolj smiselno bi bilo, če bi tu definirali optimalno hitrost $v_{opt} = 1$, ki bi zamenjala maksimalno hitrost v naši omejitveni funkciji. Tako bi prevelike in premajhne hitrosti bile bližje optimalni.

2 Fiksen robni pogoj

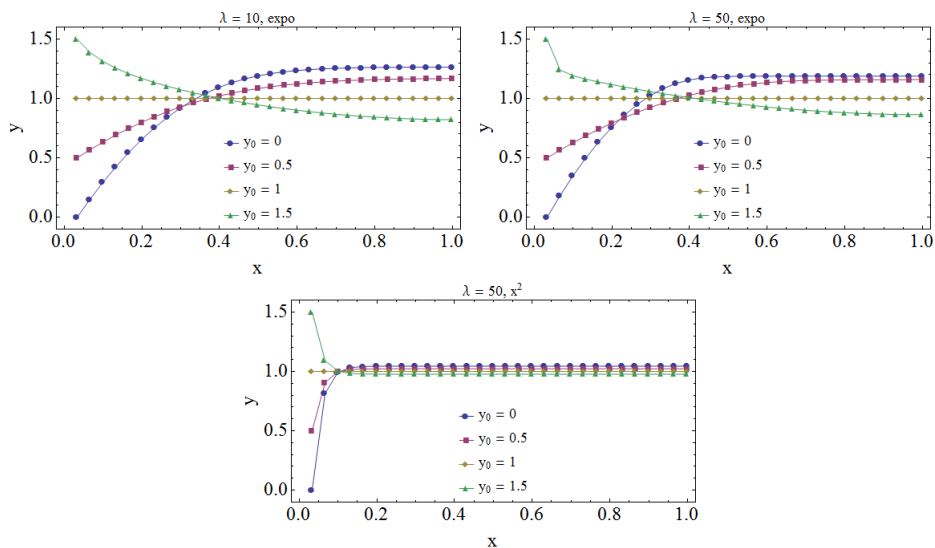
S pogojem $v_N = v_{END}$ dobimo izbrano določeno končno hitrost. Naredil sem dva primera, kjer je prvi je za končno hitrost $v_N = 3$, drugi pa za $v_N = 1$:



Vidimo, da sta prvi dve rešitvi nekoliko nesmiselni zaradi prevelike končne hitrosti, na spodnji sliki pa ima omejitvena funkcija tudi pričakovan učinek.

3 Dinamični robni pogoj

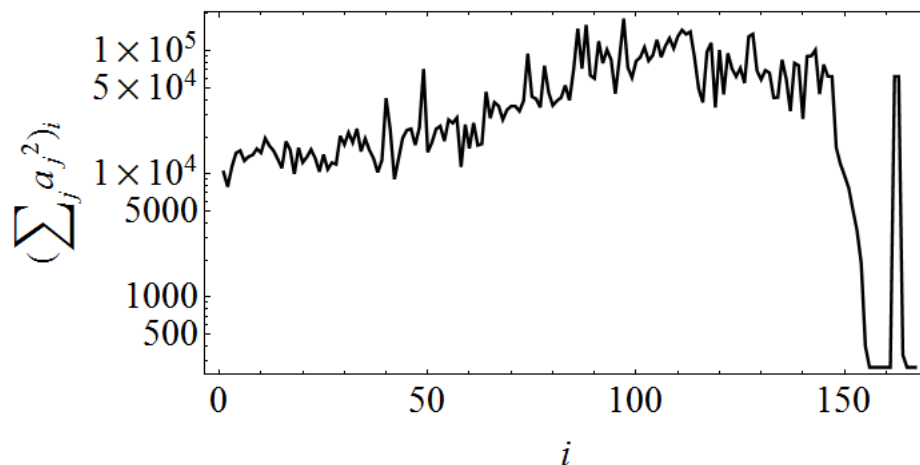
S pogojem \dot{v}_N dobimo dinamičen robni pogoj. Tudi v tem primeru so rešitve precej podobne prejšnjim. Vidimo, da omejitev funkcije povzroči, da je pospešek enak 0 že prej:



V prvih dveh primerih, kjer imamo eksponentno omejitveno funkcijo, imamo še smiselne rešitve, ki delujejo tudi precej optimalne, tudi v primeru $\lambda = 50$, medtem ko kvadratna funkcija na sliki spodaj hitro pokvari udobnost vožnje.

4 Potek minimizacije

Kot v prvem delu, si lahko tudi tu ogledamo potek minimizacije in odvisnost kumulativnega kvadrata pospeška od števila iteracij funkcije NMinimize, kjer sem uporabil kar samodejno izbrano metodo. Oglejmo si vrednost kumulativnega kvadrata pospeška v odvisnosti od števila iteracij: enak 0 že prej:



Vidimo, da na začetku vrednost količine narašča in minimizacija očitno še ne pride do izraza. Verjetno tu metoda preizkuša vse možne konfiguracije, kjer se nato odloči za optimano, kot je opaziti okoli 150. iteracije na grafu. Za lažjo predstavilo prilagam še povezavo do postopka delovanja te metode:

<https://dl.dropboxusercontent.com/u/12747812/Animacije/avto.gif>.

Na spodnji povezavi je možen še prenos .rar datoteke, ki vsebuje vse animacije tega poročila:

https://dl.dropboxusercontent.com/u/12747812/Animacije/anim_mod_3.rar.