7. naloga: Newtonov zakon

Gibanje masne točke v polju sil v eni dimenziji opišemo z diferencialno enačbo drugega reda, z Newtonovim zakonom

$$m\,\frac{\mathrm{d}^2x}{\mathrm{d}t^2} = F\,.$$

Enačba je seveda enakovredna sistemu enačb prvega reda

$$m\,\frac{\mathrm{d}x}{\mathrm{d}t} = p\,, \qquad \frac{\mathrm{d}p}{\mathrm{d}t} = F$$

in tako jo tudi rešujemo: kot sistem dveh enačb prvega reda.

Seveda morajo biti na voljo tudi ustrezni začetni pogoji, tipično $x(t=0) = x_0$ in $dx/dt = v(t=0) = v_0$. Splošnejše gre tu za sistem diferencialnih enačb drugega reda:

$$\frac{\mathrm{d}^n y}{\mathrm{d}x^n} = f(x, y, y', y'', ...),$$

ki ga lahko prevedemo na sistem enačb prvega reda z uvedbo novih spremenljivk v slogu gibalne količine pri Netwonovi enačbi ($y' = v, y'' = z, ...$).

Z nekaj truda se da eksplicitno dokazati, mi pa lahko privzamemo, da so metode za reševanje enačb hoda (Runge-Kutta 4. reda, prediktor-korektor...) neposredno uporabne za reševanje takšnih sistemov enačb in torej aplikabilne v poljubno dimenzijah, kar naj bi v principu zadovoljilo večino naših zahtev.

Obstaja še posebna kategorija tako imenovanih *simplektičnih* metod, za enačbe, kjer je f le funkcija koordinat, $f(y)$, ki (približno) ohranjajo tudi Hamiltonian, torej energijo sistema. Najbolj znana metoda je Verlet/Störmer/Encke metoda, ki je globalno natančna do drugega reda in ki točno ohranja tudi vrtilno količino sistema (če je ta v danem problemu smiselna). Rešujemo torej za vsak diskretni korak n velikosti h, $x_n = x_0 + n \cdot h$:

$$\frac{\mathrm{d}^2 y}{\mathrm{d}x^2} = f(y)$$

in pri diskretizaciji dobimo recept za korak y_n in $v_n = y'_n$:

$$\begin{aligned} y_{n+1} &= y_n + h \cdot v_n + \frac{h^2}{2} \cdot f(y_n) \\ v_{n+1} &= v_n + \frac{h}{2} \cdot [f(y_n) + f(y_{n+1})]\,. \end{aligned}$$

Alternativno lahko to shemo zapišemo tudi s pomočjo dodatnih vmesnih točk in preskakujemo med lego in hitrostjo z zamikom h/2 (od tod angleško ime 'leapfrog' za ta zapis):

$$\begin{aligned} y_{n+1} &= y_n + h \cdot v_{n+1/2} \\ v_{n+3/2} &= v_{n+1/2} + h \cdot f(y_{n+1}). \end{aligned}$$

V še enem drugačnem zapisu je metoda poznana tudi kot metoda "Središčne razlike" (Central Difference Method, CDM), če nas hitrost ne zanima:

$$y_{n+1} - 2y_n + y_{n-1} = h^2 \cdot f(y_n),$$

kjer prvo točko y_1 izračunamo po originalni shemi. Metodo CDM lahko uporabljamo tudi za primere, ko je f tudi funkcija 'časa' x, f(x,y), le da tu simplektičnost ni zagotovljena (in tudi verjetno ne relevantna). Za simplektične metode višjih redov je na voljo na primer Forest-Ruth metoda ali Position Extended Forest-Ruth Like (PEFRL) metoda, ki sta obe globalno četrtega reda in enostavni za implementacijo.

Naloga: Čim več metod uporabi za izračun nihanja matematičnega nihala z začetnim pogojem $\vartheta(0) = \vartheta_0 = 1$,$\dot{\vartheta}(0) = 0$. Poišči korak, ki zadošča za natančnost na 3 mesta. Primerjaj tudi periodično stabilnost shem: pusti, naj teče račun čez 10 ali 20 nihajev in poglej, kako se amplitude nihajev sistematično kvarijo. Pomagaš si lahko tudi tako, da občasno izračunaš energijo $E \propto 1 - \cos\vartheta + \frac{\dot{\vartheta}^2}{2\omega_0^2}$. Nariši tudi ustrezne fazne portrete!. Z analitično rešitvijo dobimo za nihajni čas $\frac{4}{\omega_0} K\left(\sin^2\frac{\vartheta_0}{2}\right)$, kjer je $K(m)$ popolni eliptični integral prve vrste, ki je v SciPy knjižnici in v članku na spletni učilnici podan z:

$$K(m) = \int_0^1 \frac{dz}{\sqrt{(1-z^2)\,(1-mz^2)}} = \int_0^{\frac{\pi}{2}} \frac{du}{\sqrt{(1-m\sin^2 u)}}$$

Previdno, obstaja tudi definicija z m^2 v integralu - potem je prav $K\left(\sin\frac{\vartheta_0}{2}\right)$, brez kvadrata (npr že v Wikipediji)!
(Dodatno lahko tudi sprogramirate eliptični integral, ki je analitična rešitev dane enačbe ali pa ga vzamete iz ustreznih programskih knjižnjic).

Dodatna naloga: Razišči še resonančno krivuljo vzbujenega dušenega matematičnega nihala

$$\frac{\mathrm{d}^2x}{\mathrm{d}t^2} + \beta\frac{\mathrm{d}x}{\mathrm{d}t} + \sin x = v\cos\omega_0 t\,,$$

kjer je β koeficient dušenja, v in ω_0 pa amplituda in frekvenca vzbujanja. Opazuj obnašanje odklonov in hitrosti nihala pri dušenju $\beta = 0.5$, vzbujevalni frekvenci $\omega_0 = 2/3$ in amplitudo vzbujanja na območju $0.5 < v < 1.5$. Poskusi opaziti histerezno obnašanje resonančne krivulje pri velikih amplitudah vzbujanja (Landau, Lifšic, CTP, Vol. 1, *Mechanics*).

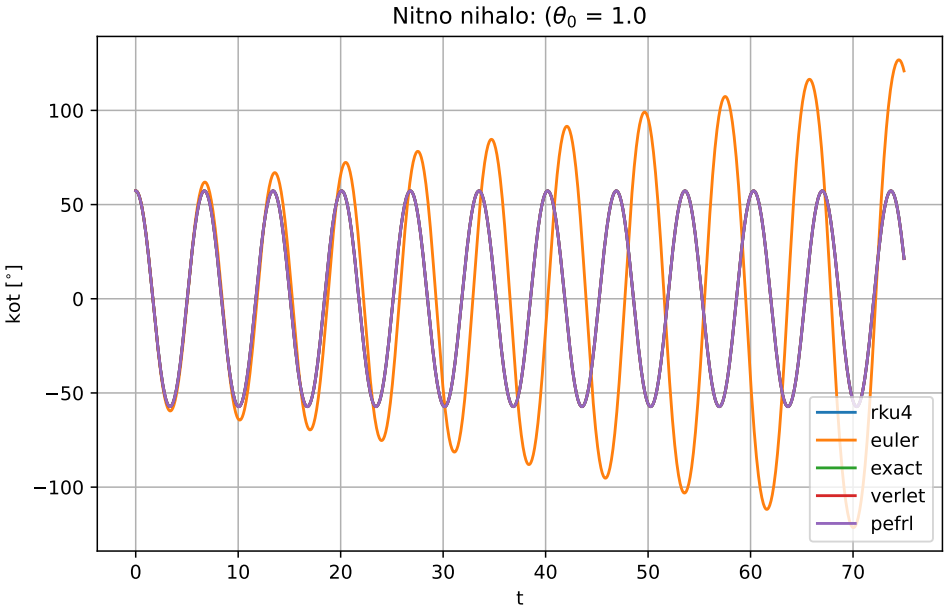
Dodatna dodatna naloga: če ti gre delo dobro od rok, si oglej še odmike in hitrosti (fazne portrete) van der Polovega oscilatorja

$$\frac{\mathrm{d}^2x}{\mathrm{d}t^2} - \lambda\frac{\mathrm{d}x}{\mathrm{d}t}\left(1-x^2\right) + x = v\cos\omega_0 t$$

s parametri $\omega_0 = 1$, $v = 10$ ter $\lambda = 1$ ali 100. Tu se ne trudi s preprostimi diferenčnimi shemami: problem je nelinearen in tog, zato uporabi neko preverjeno metodo (na primer iz družine Runge-Kutta ali ekstrapolacijsko metodo) s prilagajanjem velikosti koraka.

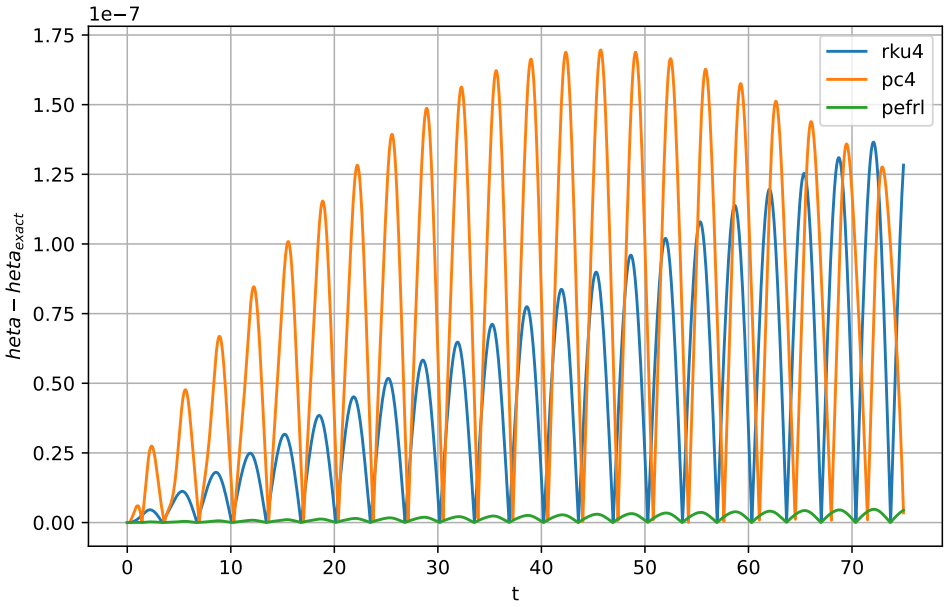
1 Reševanje

Najprej lahko narišemo čisto navadne grafe izračunov rešitve diferencialne enačbe. Eksaktna enačba je seveda dvojni eliptični integral, ki jo lahko zgolj poberemo iz knjižnice SciPy. Iz grafa lahko opazimo, da so vse metoda na prvi pogled zelo natančne (v mojem primeru majhnega δt), razen Eulerjeve metode, ki bi potrebovala še bistveno menjšega, da bi bila uporabna. Pri Eulerjevi metodi se očitno iz grafa energija pridobiva, kar je seveda fizikalno gledano malce smešno.



Slika 1: Osnoven graf nihal, z različnimi metodami

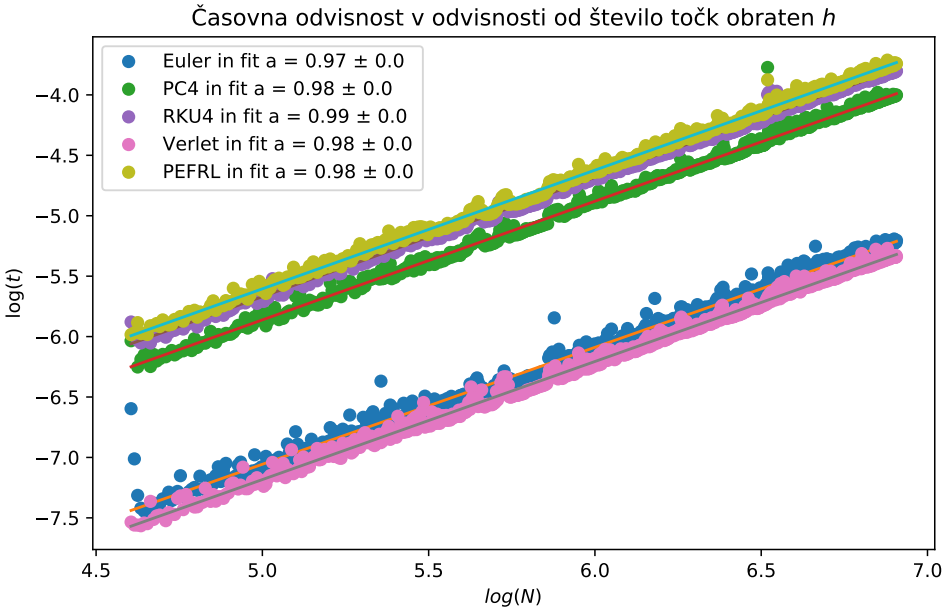
Če se sedaj osredotočimo na napako, ki jo dobimo pri metodah, je smiselno narisati časovno odvisnost napak od analitične (ki vbistvu sama po sebi ni popolnoma natančna, ker je zgolj numerično izračunan eliptični integral, vendar je zelo dobra referenca). Eulerjeve metode sploh ne bom narisal, ker ni smiselno. Vidimo, da metode lahko dosežejo zelo veliko natančnost. Problem Runga-Kutta (ki je gold standard) je, da napaka v odvisnosti od časa linearno narašča Energija pa se izgublja, kar nam zelo poslabša sliko. Boljša je simplektična metoda PEFRL, ki napako ohranja blizu ničle, Energija pa se tudi ohranja, kot bomo videli kasneje.



Slika 2: Napake, z različnimi metodami

1.1 Časovna zahtevnost

Časovno zahtevnost izračunamo identično, kot pri prejšni vaji.

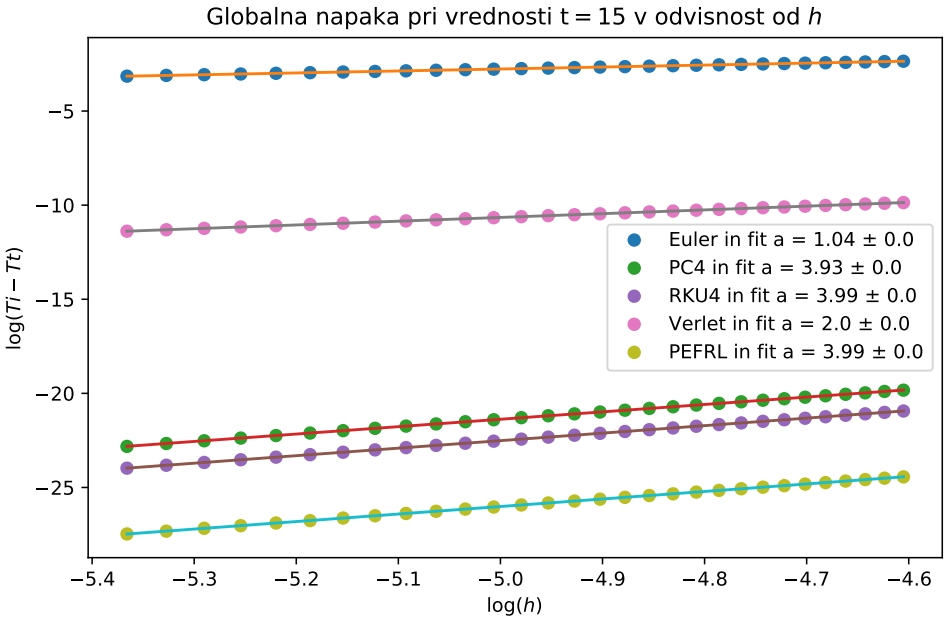


Slika 3: Časovne odvisnost vseh metod, se sklada s teorijo

Vidimo, da imajo vse metode linearno časovno zahtevnost, če zmanjšujemo h oziroma povečujemo število točk, v katerih računamo enačbo. Od vseh teh je Verlet metoda daleč najhitrejša (ker opravljamo zgolj eno operacijo).

1.2 Globalna napaka

Globalno napako bom izračunal tako, da bom na intervalu izračuna zgolj pogledal maksimalen odmik od eksaktne vrednosti. Nato bom to naredil v odvisnosti od h. Kar bi moral dobiti je, da je naprimer Eulerjeva metoda linearno natančna, RKU4 pa je natančna do polinoma 4 stopnje.

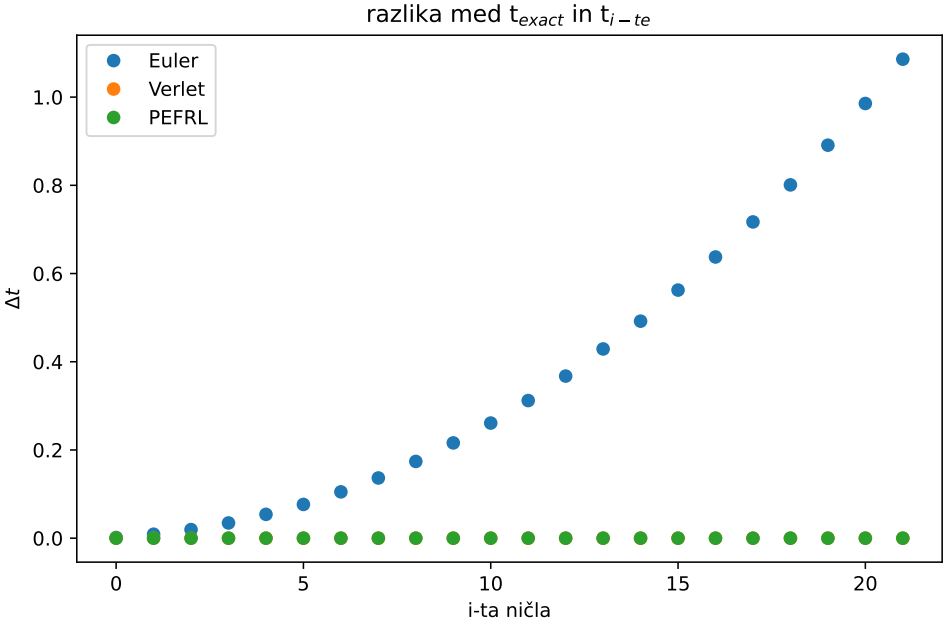


Slika 4: Tudi globalne napake metod se skladajo s teorijo

Iz logaritmiranja napak lahko ugotovimo, da se teoretične napovedi, zares strinjajo z izračunom. Torej smo potrdili, da je metoda Verlet natančna do polinoma 2 stopnje, PEFRL pa do polinoma 4 stopnje.

1.3 Ohranitev periode

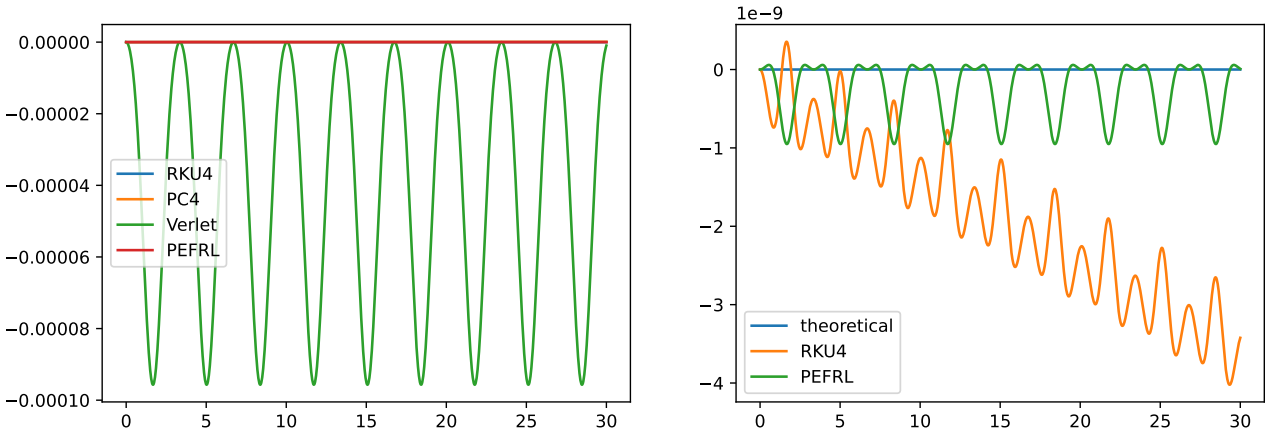
Uporabil bom funkcijo `scipy.signal.argrelextrema` za izračun maksimumov.



Slika 5: Razlika med periodami je neopazna pri zelo veliki natančnosti, razen pri Eulerjevi metodi.

Iz grafa lahko odčitamo, da se pri simplektičnih metodah zelo malo razlikjejo nihajni časi od dejanske eksaktne vrednosti - verjetno se tudi pokvarijo, vendar nisem uspel izračunati dovolj veliko točk, da bi se razlika opazila.

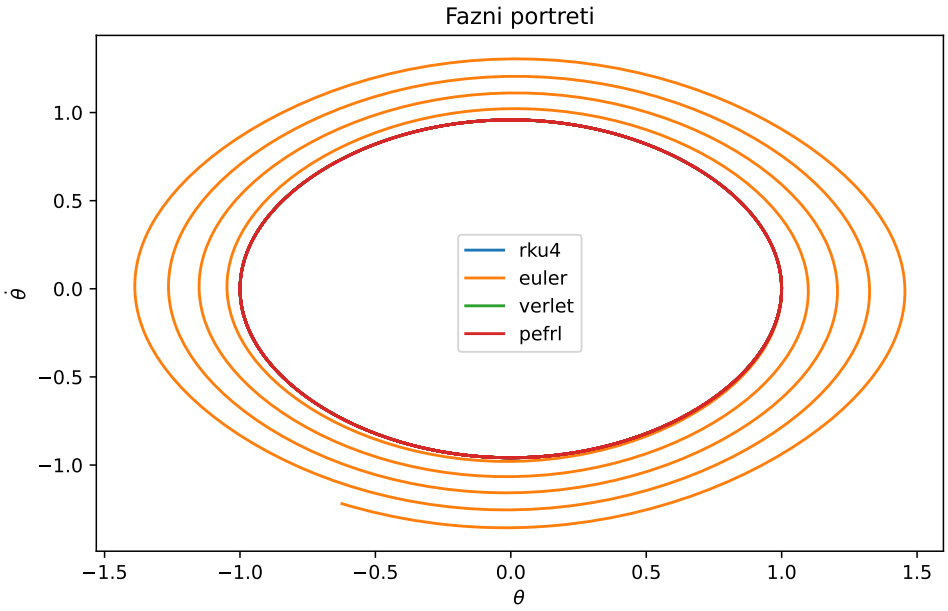
1.4 Ohranitev energije



Slika 6: Ohranitev energije posameznih metod.

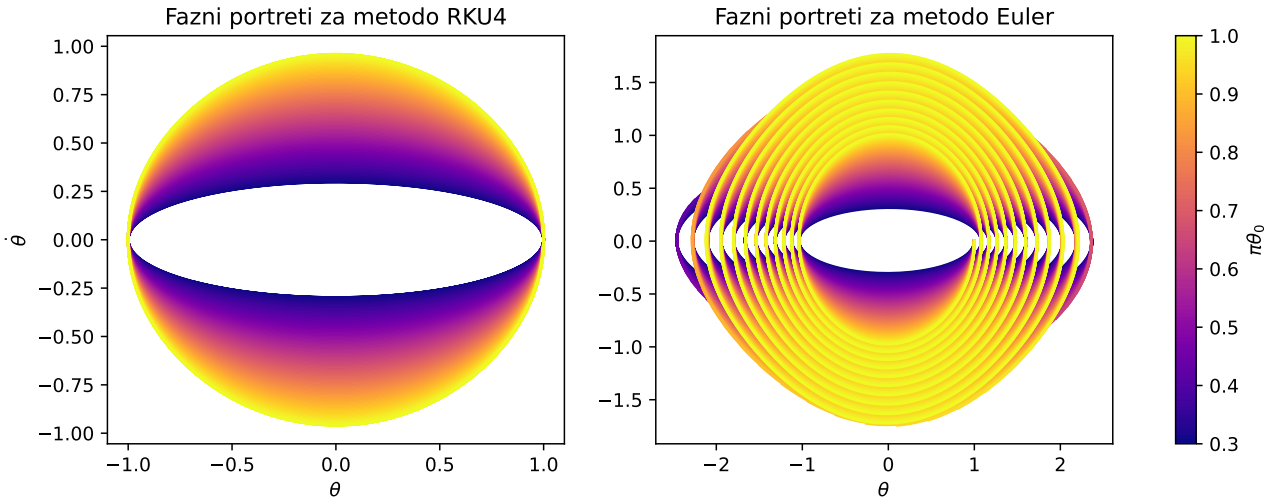
Vidimo, da so metoda iz vidika ohranitve energije zelo natančne, vendar ob bolj natančnem pogledu se opazi, da energija pri RKU4 linearno (in periodično) pada. Prednost simplektičnih metod je zelo dobro vidna na grafu, ker pri velikih časih konvergira in ne divergira.

1.5 Fazni portreti



Slika 7: Osnovni fazni portreti.

Fazni potreti vseh metod na izgled sovpadajo, ker so tako blizu. Eulerjeva metoda pride kot nekakšna spirala, ki se s časom povečuje, ker raste energija. Pri navadnem matematičnem nihalu niso portreti nič kaj preveč zanimivi, zato je za lepši izgled smiselno narisati tudi parametrizirano verzijo, kjer se začetni kot spreminja.

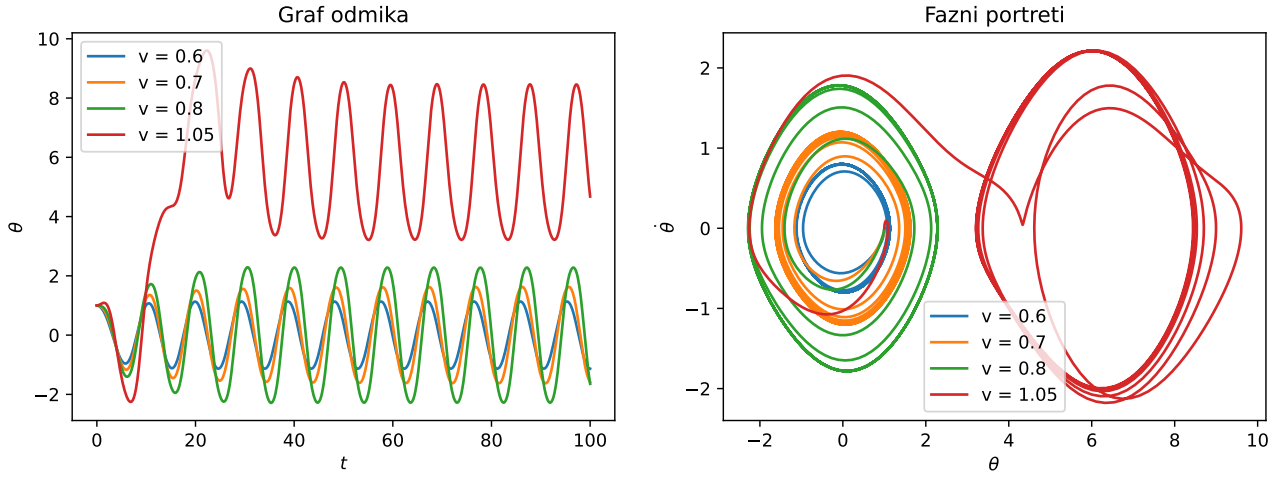


Slika 8: Lepi fazni portreti. Trdim, da so Egipčani tukaj dobili navdih za piramide.

Kar lahko opazimo iz tega grafa je, da se z manjšim kotom fazni portret oža (največja hitrost je čedalje manjša). Poleg tega lahko narišemo tudi Eulerjevo metodo, ki ima zelo unikaten in zanimiv izgled, ki bi ga verjetno lahko uporabili v kakšni Egipčanski piramidi.

1.6 Dodatna enačba #1

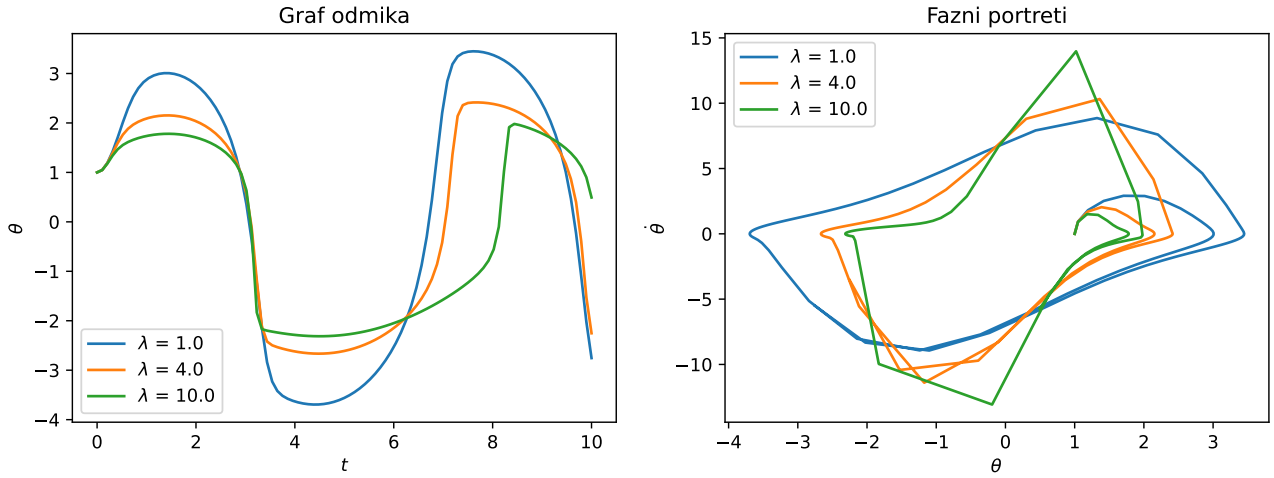
Zdaj, ko imamo za najlažji primer napisane funkcije, ni težko narediti precej bolj zakomplicirane funkcije.



Slika 9: Fazni portreti za enačbo $\frac{\mathrm{d}^2x}{\mathrm{d}t^2} + \beta\frac{\mathrm{d}x}{\mathrm{d}t} + \sin x = v\cos\omega_0 t$

1.7 Van Der Pool

Za izračun Van Der Poola sem moral uporabiti integrerano funkcijo Scipy-ja Odeint, ker ni šlo z Numpy metodami.



Slika 10: Van Der Poolov oscilator in rešitve

2 Komentar

Grafe sem dobil precje lepe in smiselne. Rad bi se zgolj opravičil še v poročilu za pozno oddajo, vendar sem se malce predolgo časa hecal, da bodo grafi lepi. Naučil sem se tudi `Manim` plugin za Python, in bom naslednič poskusil narediti kakšne lepe animacije.