Bachelor's Thesis

Playing around with parameters

Optimizer

STLSQ

"*Sequentially thresholded least squares*"

They take the argument λ, which specifies a threshold "of sparsity". Should parameter p_i be smaller than λ, it will be removed (and it's corresponding term)

Meaning that the bigger the threshold is, the **less** terms may be present in the final model. In other words, the bigger the threshold, the more terms will be removed.

For `1D-Neuron-Multiple` the value of λ seemed to be best around $(10^{-1}, 10^{-3})$, smaller values of λ included insignificant noise.

> Default value is $\lambda = 10^{-1}$

Also for the [basic example]() they set the value to $\lambda = 10^{-1}$, which seems to agree.

`TODO: STLSQ with vector of thresholds`

Metrics

L_2 Norm error

L_2 (or l_2) norm of the error gives the l_2 of an error in each dimension

Of course the smaller the better

TVDIFF

Most times around `300` iterations seems to be by far enough

Regularization α

The regularization parameter α tells us how strongly should the derivative be regularized (think of it as smoothed)

The bigger the α the less it oscillates, though the less "features" of the true derivative it really exhibits

It is mostly visible when there is a big spike in the derivative. Then the `tvdiff` is unable "to catch up" when strongly regularized doesn't handle the spike well (it simply doesn't feature nearly as big of a spike)

I'd recommend starting with a higher α and slowly increasing it, until we find the derivative smooth enough.

With small α always check if it more or less corresponds to data (it has tendency to oscillate when the function is too constant)

Epsilon ε

Using tvdiff with `ε=1e-9`, we obtain a strongly regularized result. Larger values of `ε` improve conditioning and speed, while smaller values give more accurate results with sharper jumps.

Scale and preconditioner

`TO BE DONE`

Performance

In general the more data (and thus the derivation) varies in scale, the worse to model performs

Collocations

Data collocation is only used when derivative is NOT supplied (and is surely better than forward diff)

`TODO: Usage collocations on existing derivative?`

Data.csv Structure

File `data.csv` should follow structure, where each "run" is suffixed by `__` and the index of the run. For example the columns could be

- `time__1`
- `x__1`
- `y__1`
- `time__2`
- `x__2`
- `y__2`

whaz

For simple singular trajectories, it remains to be done per [BTHS-19 - Explore how many trajectories can be used as opposed to just one]

Other implementations

Here are some implementations in other languages (and hopefully guides to use them)

- matlab
- python

Explanation of L2 norm of an error

When solving for the sparsest possible set of DEs, it is likely our found model will **not** describe the data exactly - there will be an error

Therefore we can measure the error and give the user it's l^2-norm

- The error is a vector of errors at each time-step
- More information regarding the l^2-norm is here

When working with 2 or more dimensional data, the l^2-norm returned will be vector of l^2 norms in each coordinate

Various cutoffs

Should be

```
Differential(t)(V) = p₁ + V*p₂ + W*p₄ + p₃*(V^3)
Differential(t)(W) = p₅ + V*p₆ + W*p₇
```

Always the title is `cutoff` and `optimization method`

Smooth Forward Df

2000 & STLSQ

```
Model ##Basis#388 with 2 equations
States : V W
Parameters : 6
Independent variable: t
Equations
Differential(t)(V) = p₁ + V*p₂ + W*p₄ + p₃*(V^2)
Differential(t)(W) = p₅ + W*p₆

Linear Solution with 2 equations and 6 parameters.
Returncode: solved
L₂ Norm error : [74.78388648297391, 25.370167490831182]
AIC : [34110.303086665364, 25566.935128488396]
R² : [-1.7705432321293069, 0.48830612769746606]



Parameters:
[0.3, 0.8, -0.4, -0.3, 0.6, -0.31]
```

1000 & SR3

Model ##Basis#405 with 2 equations

States : V W

Parameters : 8

Independent variable: t

Equations

Differential(t)(V) = $V*p_1 + W*p_4 + p_2*(V^2) + p_3*(V^3)$

Differential(t)(W) = $p_5 + V*p_6 + W*p_8 + p_7*(V^2)$

Linear Solution with 2 equations and 8 parameters.

Returncode: solved

L_2 Norm error : [211.63117047913556, 1062.8083759316835]

AIC : [47690.18501396972, 62058.069854531444]

R^2 : [-3.0596412951792242, 0.6042583461776001]

Parameters:

[1.09, -0.8, 0.11, -0.15, 0.24, -0.13, 0.21, -0.14]

2000 & SR3

Model ##Basis#411 with 2 equations

States : V W

Parameters : 8

Independent variable: t

Equations

Differential(t)(V) = $p_1 + V*p_2 + p_3*(V^2) + p_4*(V^3)$

Differential(t)(W) = $p_5 + V*p_6 + p_7*(V^2) + p_8*(W^2)$

Linear Solution with 2 equations and 8 parameters.

Returncode: solved

L_2 Norm error : [288.2814774371565, 1582.4585434912008]

AIC : [44778.096916964794, 58235.30636382105]

R^2 : [-9.680058684080175, -30.916791253738456]

Parameters:

[-0.5, 1.8, -1.4, 0.29, 0.27, -0.25, 0.3, -0.18]

Tvdiff Df

1000 & SR3

Model ##Basis#507 with 2 equations

States : V W

Parameters : 8

Independent variable: t

Equations

Differential(t)(V) = $V*p_1 + W*p_4 + p_2*(V^2) + p_3*(V^3)$

Differential(t)(W) = $p_5 + V*p_6 + W*p_8 + p_7*(V^2)$

Linear Solution with 2 equations and 8 parameters.

Returncode: solved

L_2 Norm error : [211.3101858827484, 1053.3627931231706]

AIC : [47676.671431685485, 61978.59180147926]

R^2 : [-3.054002039146713, 0.6099054364959031]

Parameters:

[1.09, -0.8, 0.11, -0.15, 0.24, -0.13, 0.21, -0.14]

1 & SR3

Model ##Basis#446 with 2 equations

States : V W

Parameters : p_1 p_2 p_3 p_4

Independent variable: t

Equations

Differential(t)(V) = 0

Differential(t)(W) = $p_1 + V*p_2 + W*p_4 + p_3*(V^2)$

Linear Solution with 2 equations and 4 parameters.

Returncode: solved

L_2 Norm error : [155.0710066105855, 9291.229194306132]

AIC : [49952.53064617247, 90480.8524061853]

R^2 : [-0.30329307021503427, 0.8870442179710343]

Parameters:

[0.27, -0.28, 0.26, -0.11]

With smoothing and without supplied derivative

1000 & SR3 & GaussianKernel

Notes

$$ \xdef\scal#1#2{\langle #1, #2 \rangle} \xdef\norm#1{\left\lVert #1 \right\rVert} \xdef\dist{\rho} \xdef\and{\&}\xdef\AND{\quad \and \quad}\xdef\brackets#1{\left\{ #1 \right\}} \xdef\parc#1#2{\frac {\partial #1}{\partial #2}} \xdef\mtr#1{\begin{pmatrix}#1\end{pmatrix}} \xdef\bm#1{\boldsymbol{#1}} \xdef\mcal#1{\mathcal{#1}} \xdef\vv#1{\mathbf{#1}}\xdef\vvp#1{\pmb{#1}} \xdef\ve{\varepsilon} \xdef\l{\lambda} \xdef\th{\vartheta} \xdef\a{\alpha} \xdef\vf{\varphi} \xdef\Tagged#1{(\text{#1})} \xdef\tagged*#1{\text{#1}} \xdef\tagEqHere#1#2{\href{#2\#eq-#1}{(\text{#1})}} \xdef\tagDeHere#1#2{\href{#2\#de-#1}{\text{#1}}} \xdef\tagEq#1{\href{\#eq-#1}{(\text{#1})}} \xdef\tagDe#1{\href{\#de-#1}{\text{#1}}} \xdef\T#1{\htmlId{eq-#1}{#1}} \xdef\D#1{\htmlId{de-#1}{\vv{#1}}} \xdef\conv#1{\mathrm{conv}\, #1} \xdef\cone#1{\mathrm{cone}\, #1} \xdef\aff#1{\mathrm{aff}\, #1} \xdef\lin#1{\mathrm{Lin}\, #1} \xdef\span#1{\mathrm{span}\, #1} \xdef\O{\mathcal O} \xdef\ri#1{\mathrm{ri}\, #1} \xdef\rd#1{\mathrm{r}\partial\, #1} \xdef\interior#1{\mathrm{int}\, #1} \xdef\proj{\Pi} \xdef\epi#1{\mathrm{epi}\, #1} \xdef\grad#1{\mathrm{grad}\, #1} \xdef\gradT#1{\mathrm{grad}^T #1} \xdef\gradx#1{\mathrm{grad}_x #1} \xdef\hess#1{\nabla^2\, #1} \xdef\hessx#1{\nabla^2_x #1} \xdef\jacobx#1{D_x #1} \xdef\jacob#1{D #1} \xdef\grads#1#2{\mathrm{grad}_{#1} #2} \xdef\subdif#1{\partial #1} \xdef\co#1{\mathrm{co}\, #1} \xdef\iter#1{^{[#1]}} \xdef\str{^*} \xdef\spv{\mcal V} \xdef\civ{\mcal U} \xdef\other#1{\hat{#1}} \xdef\prox{\mathrm{prox}} \xdef\sign#1{\mathrm{sign}\, #1} \xdef\brackets#1{\left(#1 \right)} $$

Počítejme $$ (Y - X \xi)^T (Y - X \xi) = \ (Y^T - \xi^T X^T) (Y - X \xi) = \ Y^T Y - Y^T X \xi - \xi^T X^T Y + \xi^T X^T X \xi $$

a pak $\grad{}$ tohoto výrazu je

$$ 0 - (Y^T X)^T - X^T Y + (X^T X + (X^T X)^T) \xi \ 0 - X^T Y - X^T Y + (X^T X + X^T X) \xi \ -2 X^T Y + 2 X^T X \xi $$

> Viz přednáška z [lineárních statistických modelů]()

Proximální operátor l_1-normy

Víme, že $\prox_ {\l \norm{\cdot}_ 1}$ je řešení minimalizačního problému $$ \min_{\vv x} \brackets{ \lambda \norm{\vv x}_1 + \frac 1 {2} \norm{\vv x - \vv v}_2^2 } $$

Uvědomme si, že $$ \grads{\vv x}{\norm{\vv x}_1} = \mtr{ \parc {|x_1|} {x_1} \ \vdots \ \parc {|x_n|} {x_n} }, $$ proto $$ \brackets{\grads{\vv x}{\norm{\vv x}_1}}_i = \begin{cases} \sign x_i & x_i \neq 0 \end{cases} $$ a také $$ \grads{\vv x}{\norm{\vv x - \vv v}_2^2} = \grads{\vv x}{\brackets{ \sum_{i = 1}^n (x_i - v_i)^2 }} = 2 \mtr{ (x_1 - v_1) \ \vdots \ (x_n - v_n) } $$ Tedy stacionární bod $\hat{\vv x}$ našeho problému musí splňovat $$ \lambda \mtr{ \parc {|x_1|} {x_1} \ \vdots \ \parc {|x_n|} {x_n} } $$

Bilevel optimization for regression of HH onto FHN

$$ \xdef\scal#1#2{\langle #1, #2 \rangle} \xdef\norm#1{\left\lVert #1 \right\rVert} \xdef\dist{\rho} \xdef\and{\&}\xdef\AND{\quad \and \quad}\xdef\brackets#1{\left\{ #1 \right\}} \xdef\parc#1#2{\frac {\partial #1}{\partial #2}} \xdef\mtr#1{\begin{pmatrix}#1\end{pmatrix}} \xdef\bm#1{\boldsymbol{#1}} \xdef\mcal#1{\mathcal{#1}} \xdef\vv#1{\mathbf{#1}}\xdef\vvp#1{\pmb{#1}} \xdef\brackets#1{\left(#1\right)}\xdef\d{\mathrm{d}} \xdef\absval#1{\left|#1\right|} \xdef\ve{\varepsilon} \xdef\l{\lambda} \xdef\th{\vartheta} \xdef\a{\alpha} \xdef\vf{\varphi} \xdef\Tagged#1{(\text{#1})} \xdef\tagged*#1{\text{#1}} \xdef\tagEqHere#1#2{\href{#2\#eq-#1}{(\text{#1})}} \xdef\tagDeHere#1#2{\href{#2\#de-#1}{\text{#1}}} \xdef\tagEq#1{\href{\#eq-#1}{(\text{#1})}} \xdef\tagDe#1{\href{\#de-#1}{\text{#1}}} \xdef\T#1{\htmlId{eq-#1}{#1}} \xdef\D#1{\htmlId{de-#1}{\vv{#1}}} \xdef\conv#1{\mathrm{conv}\, #1} \xdef\cone#1{\mathrm{cone}\, #1} \xdef\aff#1{\mathrm{aff}\, #1} \xdef\lin#1{\mathrm{Lin}\, #1} \xdef\span#1{\mathrm{span}\, #1} \xdef\O{\mathcal O} \xdef\ri#1{\mathrm{ri}\, #1} \xdef\rd#1{\mathrm{r}\partial\, #1} \xdef\interior#1{\mathrm{int}\, #1} \xdef\proj{\Pi} \xdef\epi#1{\mathrm{epi}\, #1} \xdef\grad#1{\mathrm{grad}\, #1} \xdef\gradT#1{\mathrm{grad}^T #1} \xdef\gradx#1{\mathrm{grad}_x #1} \xdef\hess#1{\nabla^2\, #1} \xdef\hessx#1{\nabla^2_x #1} \xdef\jacobx#1{D_x #1} \xdef\jacob#1{D #1} \xdef\subdif#1{\partial #1} \xdef\co#1{\mathrm{co}\, #1} \xdef\iter#1{^{[#1]}} \xdef\str{^*} \xdef\spv{\mcal V} \xdef\civ{\mcal U} \xdef\other#1{\hat{#1}} $$

Vzpomeňme si, že SINDy metoda spočívala v optimzalizaci $$ \min _{\vv \Xi} \brackets{\frac 1 2 \norm{\vv{\dot{X}} - \vv \Theta(\vv X) \vv \Xi}_2^2 + \mu R(\vv \Xi)}, $$ což můžeme v případě LASSO regrese jakožto optimalizační metody napsat jako $$ \min _{\vv \Xi} \brackets{\frac 1 2 \norm{\vv{\dot{X}} - \vv \Theta(\vv X)\vv \Xi}_2^2 + \mu \norm{\vv \Xi}_1}. $$

Předpokládejme, že máme "pevnou" trajektorii $\vv H$ HH modelu a derivace $\vv{\dot{H}}$ a obdobně pro FHN model trajektorie $\vv F$ a příslušné derivace $\vv{\dot F}$.

Potom nalezení lineární transformace $\vv \Lambda$ modelu HH na model FHN můžeme formulovat jako úlohu $$ \min _{\vv \Lambda} \underbrace{\frac 1 2 \norm{\vv F - \vv H \vv \Lambda}_2^2}_{\mcal H(\vv \Lambda)}. \tag{\T{LTR}} $$

Jak jsme si řekli, tak uvažujeme, že trajektorie $\vv H$ je "pevná", tedy že nemůžeme měnit parametry HH modelu. Naopak o modelu FHN předpokládeme, že jeho parametry měnit můžeme. Tedy bychom chtěli nalézt pro model FHN $$ \frac{\d \vv u}{\d t} = \vv f(\vv u; \vv p), $$ kde funkce $\vv f$ zadává FHN model, $\vv p$ je vektor parametrů a $\vv u$ je stav FHN systému, takové parametry, že řeší úlohu $$ \min_{\vv p} \underbrace{\frac 1 2 \sum_{i = 1}^N \norm{\vv u(t_i; \vv p) - H_{i, \cdot} \vv \Lambda}_2^2}_{\mcal F_{\vv \Lambda}(\vv p)} \tag{\T{PRM.1}}, $$ kde $H_{i, \cdot}$ je i-té pozorování HH modelu (v čase t_i) a $\vv u(t_i; \vv p)$ je pozorování FHN modelu v čase t_i za předpokladu parametrů $\vv p$. Označme $$ \vv u(T; \vv p) = \mtr{ \vv u(t_1; \vv p) \ \vdots \ \vv u(t_N; \vv p) }, \quad T = \set{t_1, \dots, t_N}. $$ Potom můžeme tyto 2 části dát dohromady a formulovat úlohu $$ \min_{\vv \Lambda} \overbrace{\frac 1 2 \norm{\vv u(T; \vv{\hat p}) - \vv H \vv \Lambda}_2^2}^{\mcal H_{\vv{\hat p}}(\vv \Lambda)} \ \text{za podmínky } \vv{\hat p} = \argmin_{\vv p} \frac 1 2 \sum_{i = 1}^N \norm{\vv u(t_i; \vv p) - H_{i, \cdot} \vv \Lambda}_2^2 \tag{\T{F2H.1}} $$ nebo také zkrácené $$ \min_{\vv \Lambda} \mcal H_{\vv{\hat p}} (\vv \Lambda) \ \text{za podmínky } \vv{\hat p} = \argmin_{\vv p} \mcal F_{\vv \Lambda}(\vv p) $$

> “ Pravděpodobně nemusíme řešit případ $\vv \Lambda = \vv 0, \vv{\hat{p}} = \vv 0$, neboť i pro $\vv{\hat p} = \vv 0$ nepovoluje tvar FHN modelu konstantní nulové řešení, které by bylo best fitem pro $\vv \Lambda = \vv 0$.

Ačkoliv by tento přístup byl jistě užitečný, dostáváme me do problému s nalezením podmínky stacionarity pro optimalizaci $$ \vv{\hat p} = \argmin_{\vv p} \mcal F_{\vv \Lambda}(\vv p) $$ Pokud bychom ji chtěli najít, museli bychom spočítat $\frac {\partial \mcal F_{\vv \Lambda}}{\partial \vv p}$, avšak bez této podmínky nejsme schopni optimalizovat celou úlohu.

> “ Toto platí v příadě, že bychom úlohu optimalizovali metodou vyžadující gradient účelové funkce. Např. metoda "Nelder-Mead" se bez něj obejde

Proto raději použijme analogii SINDy metody, která nám umožní tento problém obejít. Proto místo úlohy $\tagEq{PRM.1}$ řešme $$ \min_{\vv p} \underbrace{\frac 1 2 \sum_{i = 1}^N \norm{\vv f(\vv H_{i, \cdot} \vv \Lambda; \vv p) - \vv{\dot{H}_{i, \cdot}} \vv \Lambda}_2^2}_{\mcal F'_{\vv \Lambda}(\vv p)}, \tag{\T{PRM.2}} $$