

# Laborjournal - 1 -

09. Juli 2014

Christopher Schölzel

C. Schölzel

18. Jul. 2014

Fix für Variable  $S_i$  in shm-1 Modell

Variable  $S_i$  wurde in jedem Zeitschritt aktualisiert statt nur bei einem neuen Herzschlag.

broken  $S_i$

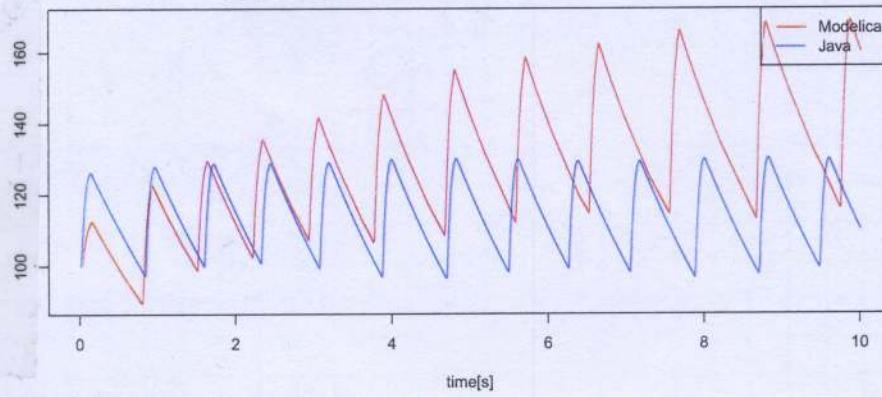
Plots vor dem Fix: .../shm1-vali/plots/brokenSi/compare.pdf  
compare-xxx.pdf

fixed  $S_i$

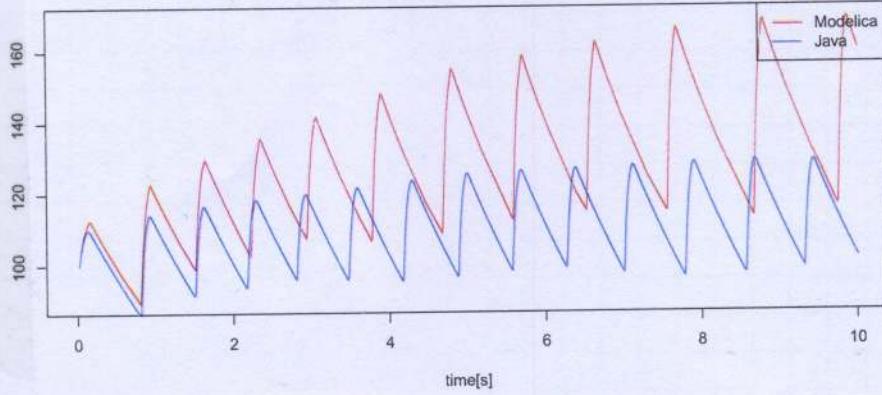
Plots nach dem Fix: .../shm1-vali/plots/fixedSi/compare.pdf  
compare-xxx.pdf

Beobachtung: Merkliche Veränderung vor allem zu Beginn der Simulation (Startparameter  $\forall S_i$  werden vor dem Fix quasi ignoriert).

broken



fixed



Vergleichsplot Java vs Modelica

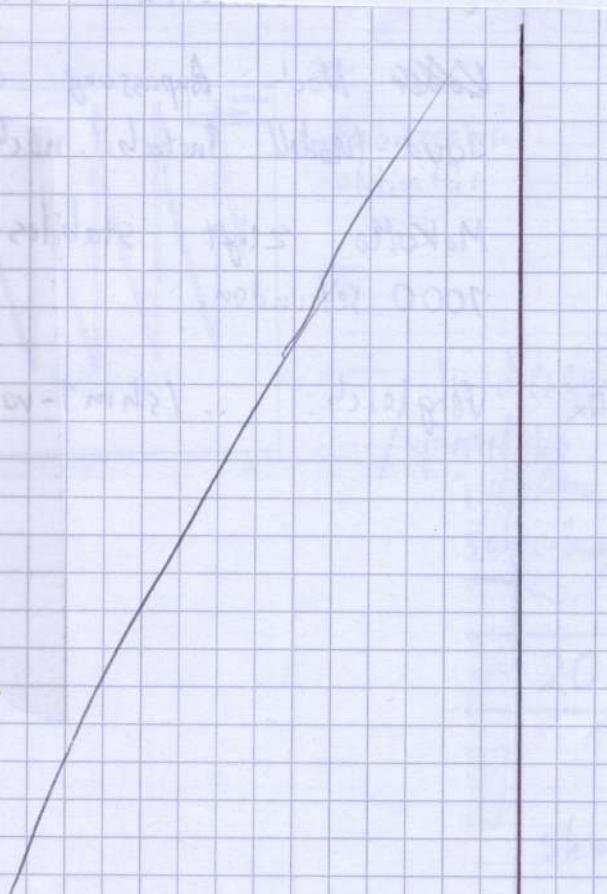
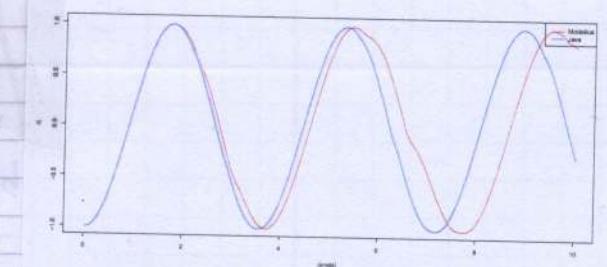
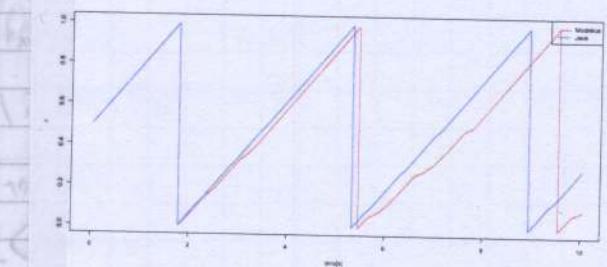
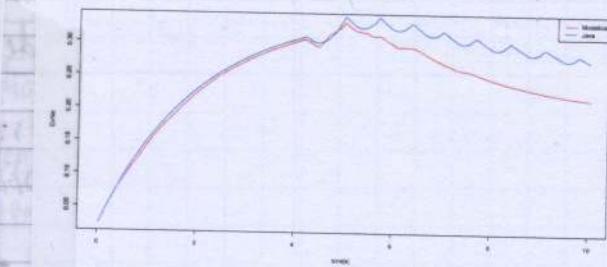
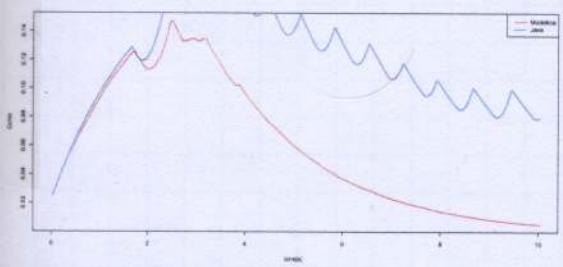
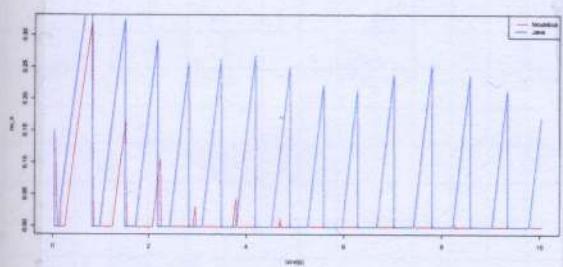
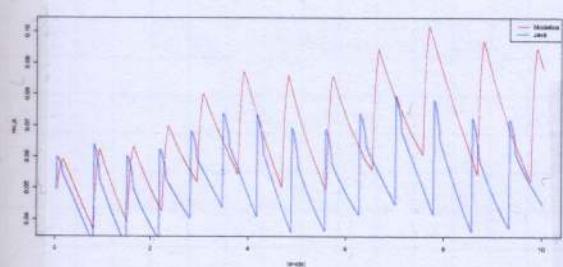
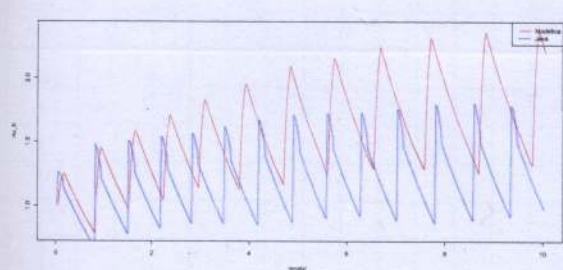
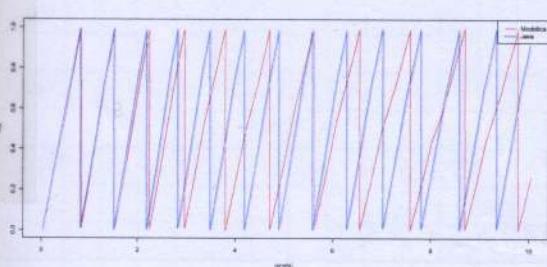
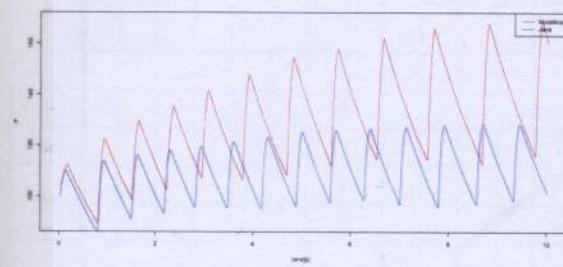
R-Script zum Vergleich von Kontakt-daten der Java - Implementierung und der Modelica - Implementierung:

plotCompare.R

.../shm1-vali/src/rscripts/plotCompare.R

⇒ Unterschiede ab der ersten Sekunde, Startwerte jetzt aber identisch.

Vor allem bei Sympathicus und Konzentrationen charakteristische Unterschiede.



Unterschiede zum Modell vor/nach Verschieben der Parameter ins Hauptmodell 29. Juli 2014

Verschieben der Parameter hatte tatsächlich doch keine funktionale Äußerung zur Folge.

Einziger Unterschied zu vorher: heart.sat ExpS = 2,5 statt 2,0

..\\Promotion\\modelica\\Resultate\\pre-post\_mainparams.ods

pre-post\_mainparams.ods

(3)

## Diagnose von Modelica-Modellen

Zusätzliche Diagnose-Outputs lassen sich mit der Funktion `Modelica.Utilities.Streams.print(<String>, <filename>)` erzeugen.

Diese muss in einem `when`-Block aufgerufen werden.

("`when initial()`" scheint seltsamerweise nicht zu funktionieren.) Bsp.: `.. /kotani /src /scripts/print-parameters.tmpl`

Eine vollständige Parameterliste erhält man durch parsen der Datei `<Modellname>.info.xml`

30. Juli 2014

Qualitative Übereinstimmung zw. Java- und Modelica-Modell

Nach Bugfix in Mokomo (Modelica - Kotani-Modell) verhalten sich alle Kurven in beiden Modellen gleich.

(Barorezeptoraktivität wurde falsch berechnet.)

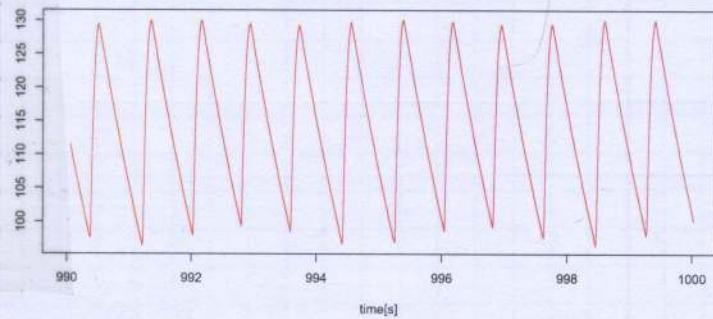
~~Korrektur~~ Nach Anpassung der Startparameter wird das Java-Modell instabil. nach ca. 4 Sekunden.

Mokomo zeigt stabiles Verhalten auch nach 1000 Sekunden.

OSO.javaInitialBaroFix

Vergleich: `.. /shm1-validated/plots/OSO.javaInitialBaroFix/compose.pdf`

↙ plot nach 1000 sek



`.. /kotani /plots /000-1000 sek-stable`

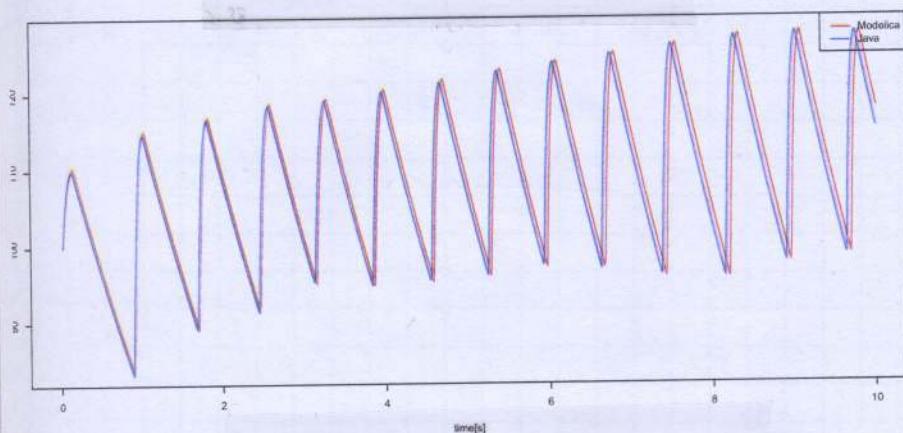
31. Juli 2014

Vollständige Übereinstimmung zw. SHM1 und Mokomo

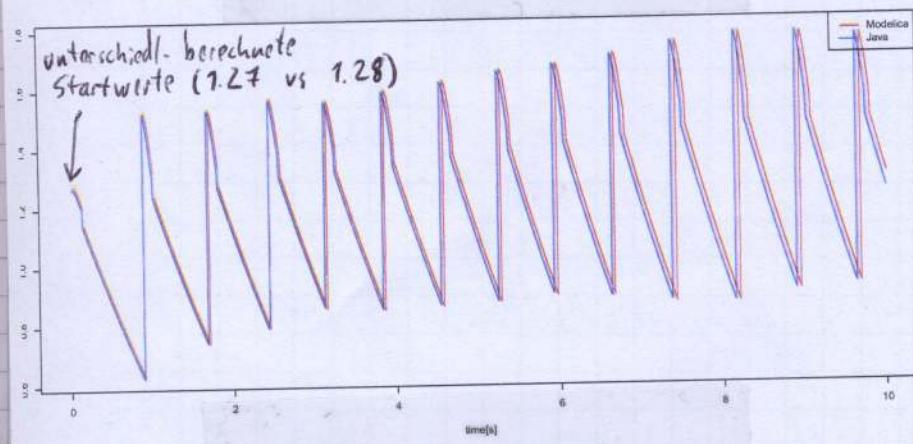
Fix der Kontraktilität in SHM1 vom 18.07.2014 war fehlerhaft  $\Rightarrow$  Noradrenalkonzentrationen waren bei Berechnung immer auf 0  $\Rightarrow$  Abfallender Blutdruck trotz steigender Sympathicus-Aktivität.

Nach der Korrektur im SHM1 verhalten sich beide Modelle bei gleichen Startparametern (Barorezeptoraktivität musste im SHM1 nochmal anders berechnet werden) genau gleich.

Vergleichsplots: ..\shm1-vali\plots\060FixedFixedSi\compare.pdf 060FixedFixedSi;



← Blutdruck



← Barorezeptoraktivität

In diesem Zeitraum:  
Erstellung einer  
detaillierten Projekt-  
beschreibung. ⇒ nosys.pdf

Analyse der Thesis von Dr. Seidel

20. Februar 2015

Gegenüberstellung aller Formeln des Seidel-Herzel Modells erstellt: shm-formulas.pdf

Vergleicht folgende Versionen:

- Seidel-Herzel 1998 (Paper)

- Kotani 2005

- Seidel 1997 (Thesis) ←

Modell ist neuer  
als in der  
1998er Version

5

20. Februar 2015 Implementierung der Barorezeptorgleichungen aus Seidels Thesis

TBaroReceptors.mo Neue Barorezeptorgleichung mit Sättigung: TBaroReceptors

Test mit Trapezoidalem Signal lässt sich nicht ~~mit~~ deckungsgleich mit den Diagrammen aus Seidels Thesis bringen  $\Rightarrow$  Vermutung: Diagramm verwendet anderen Wert für  $k_b$

Idee zur Implementierung der Broadening-Funktion:  
history-Array mit ~~verschieden~~ steigenden delay-Werten schon in TBaroReceptors definieren und an Green's Function übergeben.

23. Februar 2015 Komplettes Refactoring der Ordnerstruktur des Modelica Modells zu

MokoMo  $\rightarrow$  MoST

Neue Ordnerstruktur bietet bessere Möglichkeit, verschiedene Versionen des selben Modells zu speichern.  $\Rightarrow$  Aufteilung in

- SH7998
- Kotani 2005
- Seidel Thesis

Außerdem:

- Dokumentation angefügt (TODO: doppelte annotations)
- Saturation-Modell hat jetzt ordentliche Parameter

29. Februar 2015 Fertigstellung von Barorezeptoren, Lunge und ANS

Broadening-Funktion in Barorezeptor eingebaut

$\Rightarrow$  Problem: Mit baro\_resolution = 100 geht alles glatt, bei 2000 hört der Compiler nicht mehr auf  $\Rightarrow$  nichtlinearer Komplexitätsanstieg

Vermutung: GUI-Elemente (checkboxen) von OpenModelica Hoffnung sind das Problem.

Lunge und ANS funktionieren problemlos

Fertigstellung von Sinusknoten mit Phase Effectiveness

27. Februar 2015

Sinus node funktioniert einwandfrei.

Herz macht aber Probleme: Kontraktionsmodell ist komplizierter geworden (AV-knoten + refractory period).

a: Open Modelica: no support for discrete equation systems

Fertigstellung von Seidel Thesis

1. März 2015

Herz hat jetzt eigenes Kontraktionsmodell mit zusätzlichen kontinuierlichen Variablen als Ersatz für die diskreten.

Modell an Marek Matěják (Prag) und Alexandra Mehlhase (Berlin) geschickt zum Feedback.

Seidel Thesis wird Dymola-konform

16. März 2015

Auf erstes Feedback von Marek mussten einige Komponenten für Dymola angepasst werden:

- konditionelle Komponenten werden jetzt nur noch in Definition referenziert
- Contraction-Modell verwendet kein elsewhen mehr

Außerdem sind grafische Annotatationen jetzt auch in Seidel Thesis enthalten.

Aktualisierte Version an Marek und Alexandra geschickt.

22. April 2015

SeidelThesis stimmt mit Original überein.

Bugfixes:

- Fehlende Klammern in Lungen signal
- Modell startet jetzt mit Sinussignal statt direkt mit Kontraktion.
- Sympathikus und Parasympatikus -Aktivität darf nicht unter 0 fallen.
- heart\_p-wind0 existiert in Seidels code gar nicht  $\Rightarrow$  wurde auf 0 gesetzt.

~~Abbildung Weiterentwicklung der Plots~~

Simulation mit 7000 s erstellt:

... /SHM/output / SHM-M vs SHM-C 7000s

20. Mai 2015 Fertigstellung des Papers für die Modelica-Konferenz

Zusätzliche Funktionalität der Plot-Scripte:

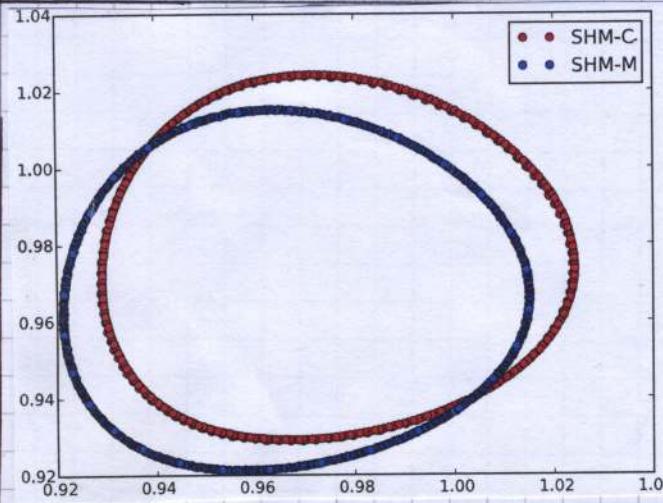
- Frequenzplots
- Plots mit Gap
- Differenzplots

SeidelThesis gibt jetzt auch Datei mit RR-Intervallen aus.

Erkenntnisse:

- RSA - Peak in SHM-M steiler
- SHM-M hat im Schnitt 3 ms längere Herzschläge

15. Juli 2015 Poincaré-Plots von SHM-M und SHM-C



projekte/paincaré compare

Plots liegen quasi übereinander, wenn die RR-Intervalle von SHM-M um 8,5 ms erhöht werden.

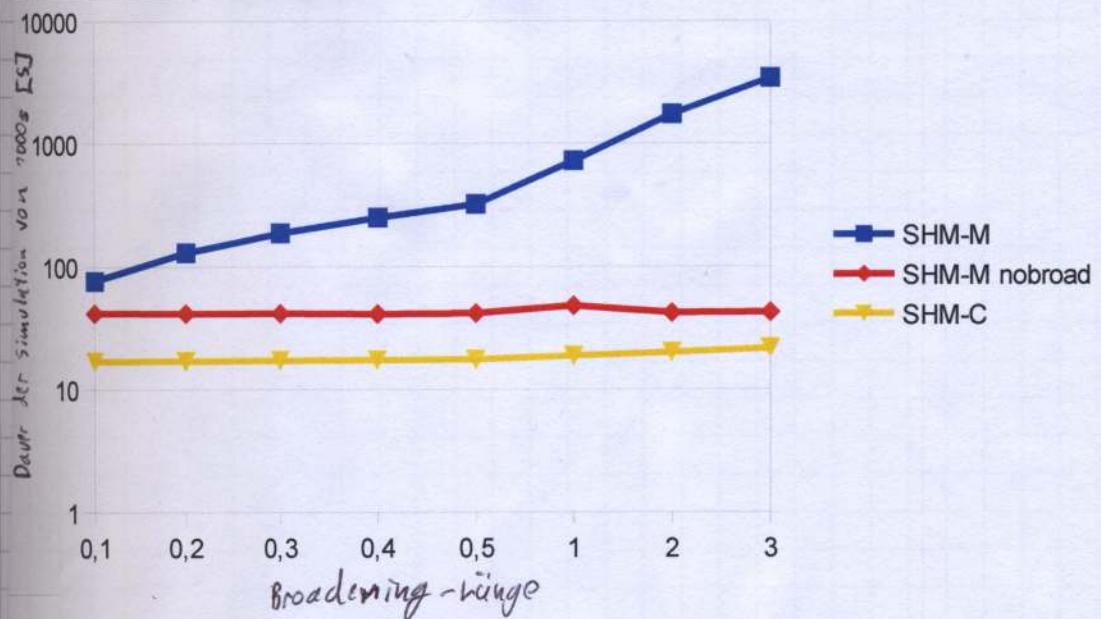
Eigentl. widersprüchlich zur Beobachtung, dass SHM-M 3 ms längere Herzschläge zu Beginn produziert.

Idee: Unterschiedliche Poincaré-Plots für leicht abgeänderte Parameter ("Bifurcationen") erstellen. Wie stabil ist das Bild überhaupt?

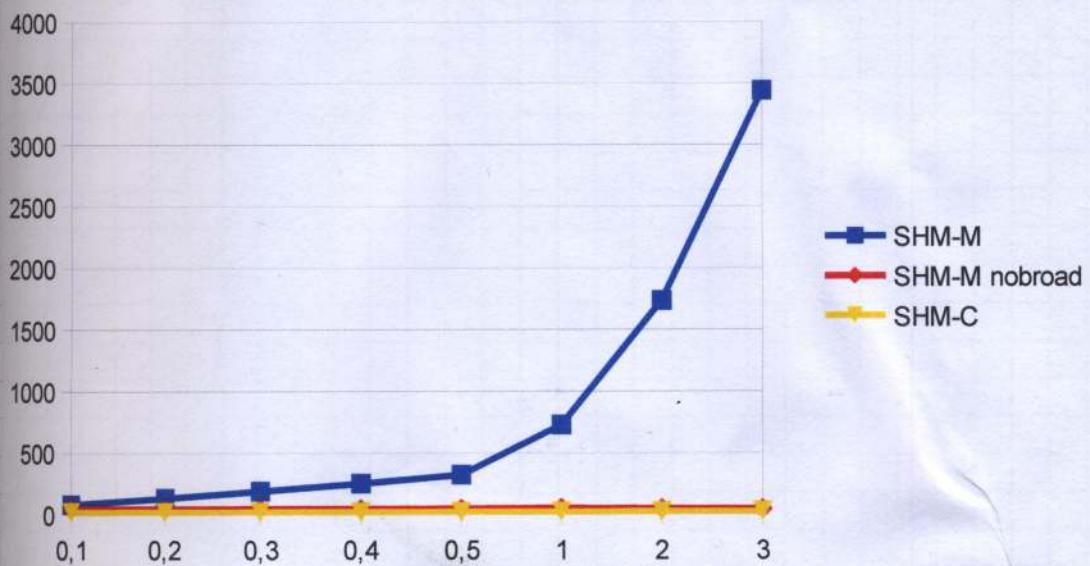
Anmerkung: Mittelmaß von RR-Intervall-Länge sagt nicht viel aus, da Punkte auf Poincaré-Plot mit großem Abstand "im Viereck springen" (wegen RSA).

## Performanceanalyse der Broadening-Funktion bei SHM-M

27. Juli 2015



One Modelica / workspace  
1 SHM / output /  
Performance



Komplexität von SHM-M scheint ~~polynomial~~ zu steigen.  
⇒ Lösung mit expliziten delay-Angaben ist extrem inefizient.

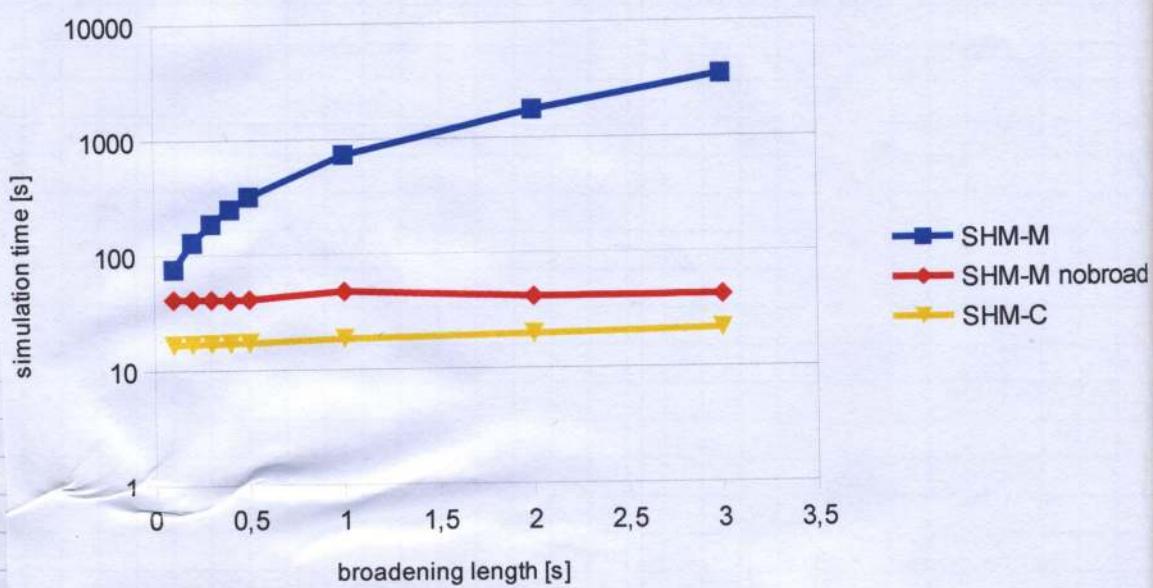
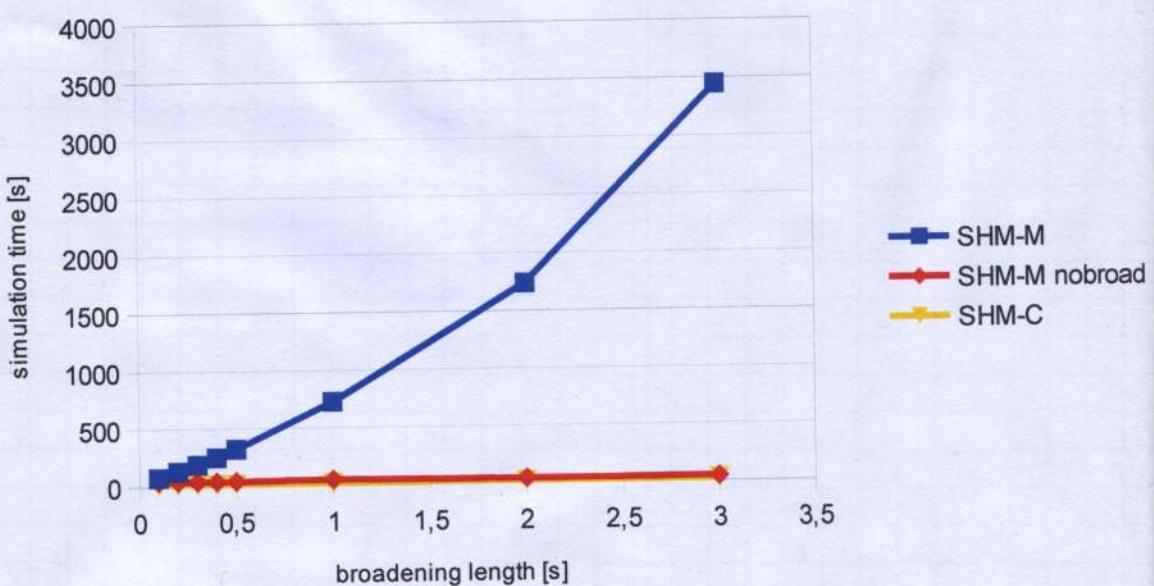
Simulation mit broad-len > 3.0 dauert 9 Stunden.

9

# 29. Juli 2015 Korrektur zur Performanceanalyse

"Polynomielles" Verhalten ist doch eher linear. LibreOffice hatte x-Achse einheitlich dargestellt trotz uneinheitlicher Messpunkte. Alle anderen Beobachtungen bleiben aber bestehen.

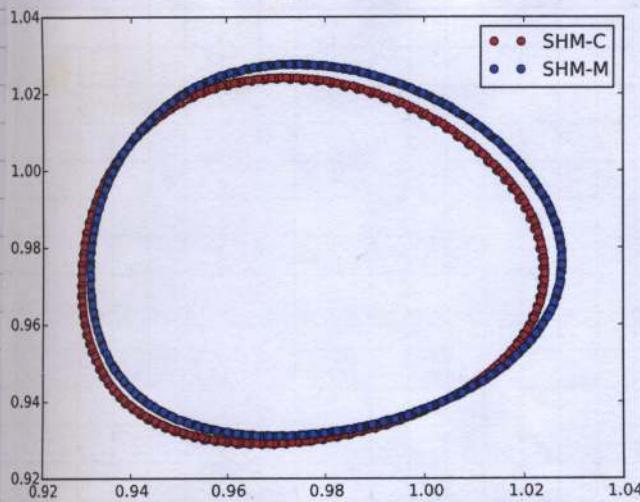
broad_len	SHM-M	SHM-M nobro	SHM-C
0,1	75,8433	41,1666	16,63
0,2	128,69	41,0582	16,74
0,3	186,3	40,8652	16,91
0,4	248,578	40,5128	17,02
0,5	320,048	41,0067	17,16
1	724,216	47,3584	18,42
2	1735,71	41,4968	19,64
3	3440,2	42,0397	21,28



## Bugfix zu fälschlicherweise zurückgesetzten Parametern in Modelica

Bei der Umbenennung von "rNe" zu "wNe" wurden zwei Parameter auf ihren Ursprungswert zurückgesetzt.

Nach der Korrektur sind die Poincaré-Plots wieder konsistent zu den anderen Simulationsergebnissen (SAM-M hat im Schnitt 3ms längere Herzperioden).



## Zusammenfassung der Konferenzen in Nizza u. Versailles

24. September 2015

### CinC 2015 (Nizza):

- hot topic: atrial fibrillation
- SVMs ~~verdeutlichen~~ häufig benutzt, selten mit viel Sinn
- Interesse an neuen Methoden zur Herzmodellierung ist vorhanden
- "Multiscale modeling" kann verschiedene Dinge heißen
  - Auffassung auf der Konferenz: Bottom-up bis zur nächsten "scale" statt paralleles Betrachten unterschiedlicher "scales"
- Nichtinvasive Messungen von Parametern außer Atmung u. Blutdruck kaum mögl. (höchstens über längere Zeitabschnitte geschützt)
- Relative RR-Intervalle sind vmtl. in vielen Fällen sinnvoller

.. /Promotion/Konferenz  
/CinC 2015 ..

### Modelica 2015 (Versailles):

- Probleme für die Systembiologie:
- kein Ausnutzen von sich wiederholenden Strukturen (arrays, loops)
- Lösung: extern C
- DASSL hat Probleme bei häufigen Events
  - Lösung: QSS 2
- "Optimierung" umfasst bisher nicht Anpassung an Messergebnisse
- Optimierung nutzt Gradientenbasierete Methoden
  - Lösung: externe Tools (Python, LabView)
- PDEs not supported

.. /Promotion/Konferenz  
/Modelica 2015 ..

11

- solution: solving / discretization by hand
- keine stoch. Modellierung (Monte Carlo)
- Modelica ist n. leicht zu lernen
- keine interaktive Simulation
- Systembiologie ~~wurde~~ weckt bereits Interesse
- Kritik am "trial and error" der Pharmaindustrie
- Tipps ~~www~~:  
- "Hum Mod" - Gruppe auf zotero.org

Eigene Idee: Interface-Library für physiologische Modelle bauen

15.12.2015

Experiment mit OMPython und C-Funktionen in Modelica

OMPython:

- Kommunikation über CORBA
  - Requirements: Python 2.7 64 bit, omniORBpy 3.0
  - PATH: omniORBpy binaries
  - PYTHONPATH: lib\python, lib\x86-win32
  - sendet String-Commands an Compiler-Shell
  - Rückgabewerte übersetzt in Python-Datenstrukturen
- ⇒ Schwerfällige Kommunikation, aber scripting möglich  
z.B. für GA-Optimierung

C-Funktionen:

- mit 'external "C" r = f(x);' definiert
  - mit 'annotation (include = "...");' direkt als String eingeschlossen
- ⇒ sehr leichte Definition bei Funktionen

19. 1. 2016

Parameterübersicht vom Seidel-Herzel-Modell

Vollständige Liste aller Parameter mit:

- Name in MoSTMo
- Name in meinen LaTeX-Dokumenten
- Name in Seidels C-Code
- Name in Seidels LaTeX-Dokumenten
- Wert laut Seidel 1997
- Einheit laut Seidel 1997

Ziel: Festlegung von Einheiten und physiologisch unveränderlichen Werten (mit Referenz zur Quelle) bei möglichst vielen Parametern.

⇒ Reduktion der freien Parameter des Modells

Promotion/Notizen/  
Parameter.xls

Beginn der Recherche zur Physiologie des menschl. Herzens

20.01.2016

Hauptquelle: „Cardiovascular Physiology Concepts“ von Richard E. Klabunde

Ziel: Breiterer Überblick um physiologische Plausibilität des NoSTMo besser ~~zu~~ beurteilen zu können.

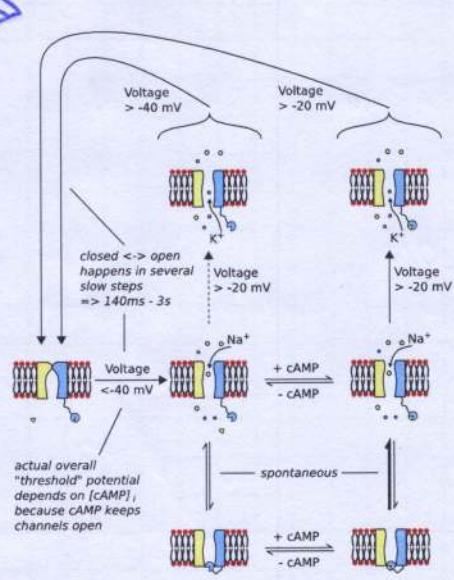
Ergebnisse werden in Docear-Mindmap festgehalten.

Eigene Schaubilder zu Ionenkanälen

03.02.2016

Erstellt wurden Vektogramm - Schaubilder für:

- schnelle Natriumkanäle
- HCN-Kanäle (hyperpolarization-activated, cyclic nucleotide gated)
- L-Type  $\text{Ca}^{++}$  Kanäle
- T-Type  $\text{Ca}^{++}$  Kanäle
- Inward rectifier
- transient outward channels
- rapid delayed rectifier
- ultrarapid delayed rectifier
- leak-channels ( $\text{K}^{+}$ )
- $\text{Na}^{+}/\text{K}^{+}$  ATPase
- Sodium-calcium exchanger
- Calcium-activated channels
- Ach-activated channels
- ATP-sensitive channels



Promotion / Notizen / Heart / ~~figures~~ images / ...

JavaScript-basierte Animation von Aktionspotentialen

13.02.2016

Animation der o.g. Ionenkanäle u. Transportmechanismen beim Aktionspotential von Schrittmacher- und Nichtschrittmacherzellen

Framework: Snap.svg

Zeitachse lässt sich mit Slider manuell verschieben, aber auch automatisch mittels Play-Button.

Vorausgeschaut werden das öffnen u. schließen der Ionenkanäle während des AP sowie der Ionentransport durch die Membran.

Promotion / Notizen / Heart / ~~figures~~ images / animations / ...

22.02.2016

Entfernen von Snap.svg - Komponenten aus AP-Anim

Beg

Switch von Snap.svg zu SVG-DOM-API  $\Rightarrow$  AP-Anim ist jetzt echtzeitfähig

10.03.2016

~~A~~ Abschluss der Recherche zur Physiologie des menschl. Herzens

Übersicht der Ergebnisse in D:\Dokumente\Promotion\Notizen\Heart\Heart.mm

Erste Erkenntnis: Barorezeptoren befinden sich hauptsächl. am Aortenbogen und Karotid sinus  $\Rightarrow$  Delay kann berechnet werden?  
 $\Rightarrow$  Broadening ist ertl. unnötig?

Erste Version eines Test-scripts für MoSTMo

Python-Unitest mit Modul "unittest" und OMPython.

Erlaubt automatisierte Tests von Simulationsdaten.

Die Modelica/  
workspace/src/  
tests/...  $\Rightarrow$  Nach jeder "Verbesserung" kann sofort geprüft werden ob Parameter immer noch in physiologisch plausiblen Grenzen liegen.

Bisher einziger Test: Simulation läuft ohne Fehler

31.03.2016

Erweiterung der Test-Suite für MoSTMo

Implementierte Tests:

- Blutdruck Min / Max / Mean / Std
- Herzrate Min / Max / Mean / Std
- Histogrammvergleich für Blutdruck u. Herzrate (mit Plots)
- Histogramm des Frequenzspektrums der Herzrate (nur Plot bisher)

Geplante weitere Tests:

- Poincaré-Parameter
- Fraktale Eigenschaften

07.04.2016

Fertigstellung der FFT-Tests für MoSTMo

Te

Testsuite enthält jetzt Test für RMSE der RR-intervall spez. denst und FFT-Plots wurden um erwartetes Ergebnis erweitert.

Re

Beginn Physionet-Challenge 2016: Klassifizierung von PCGs

03.04.2016

Ziel der Challenge : Klassifizierung von Phonocardiogrammen (abnormal / normal)

REPO: 2076-thomas-  
RWS 3016

branch: pcg2 ecf

Erster Test von Ansatz ① mit Pybrain:

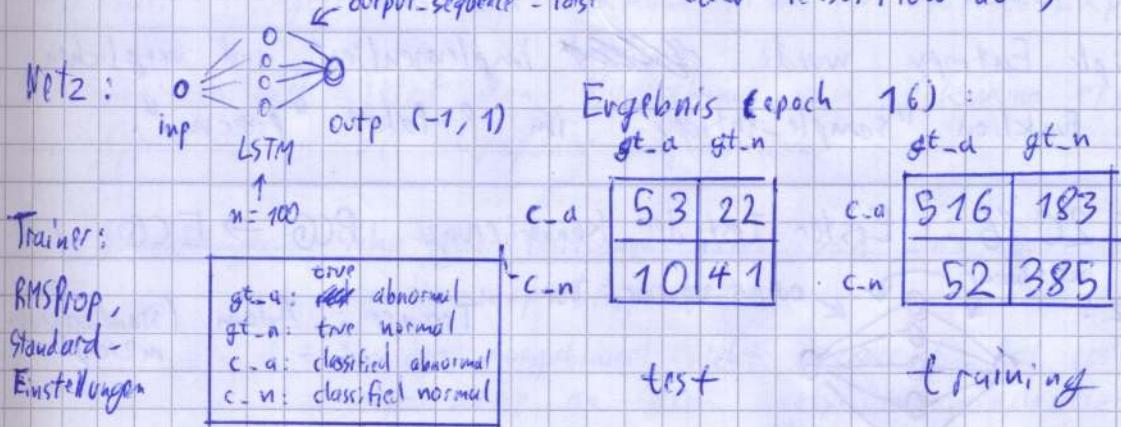
schlägt fehl, weil Pybrain Datensätze dieser Größe nicht verarbeiten kann.

⇒ Andere Framework? (OCRopus, Keras, RNNlib...)

PNC2016: Ansatz ① funktioniert mit Keras

07.04.2016

Wechsel des Frameworks: Keras (baut auf Tensor-Library Theano  
„output-sequence = Folge“ oder TensorFlow auf)



2016-thomae-puc2016  
data/results/1stfm-100-  
middle

Daten: Alle 631 normalen samples  
+ zufällige Auswahl von  
631 abnormalen samples.

## Ergenesis (epoch 18):

	$gt\_a$	$gt\_n$		$gt\_a$	$gt\_n$
$c\_a$	49	9	$c\_a$	467	100
$c\_n$	14	54	$c\_n$	701	468

Teilung training/test: 90% / 10%

Reihenfolge beim Training: abwechselnd normal/abnormal



Evolution of accuracy

### Erweiterung der Test-Suite: Sample Entropy

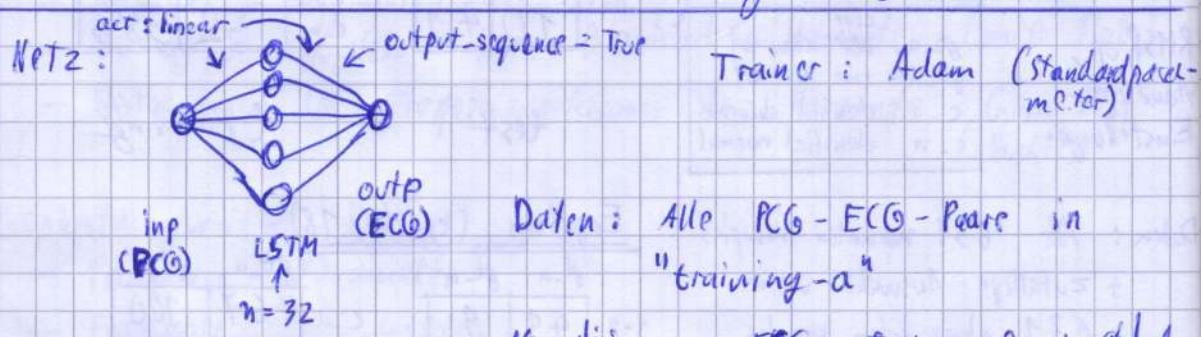
Paper of Hoshi et al. (2013) mentions several nonlinear HRV-Parameters:

- Sample Entropy (measures system complexity and unpredictability)
- Lyapunov Exponent (measures "chaos"/rate of separation of trajectories in phase space)
- Hurst Exponent (characterizes "long-term memory")
- Correlation Dimension (quantifies complexity)
- Detrended Fluctuation Analysis (similar to Hurst Exponent without assumptions on stationarity)

Sample Entropy wurde ~~effizient~~ implementiert und verglichen mit Funktion "sample\_entropy" im R-Paket "precrma".

12.04.2016

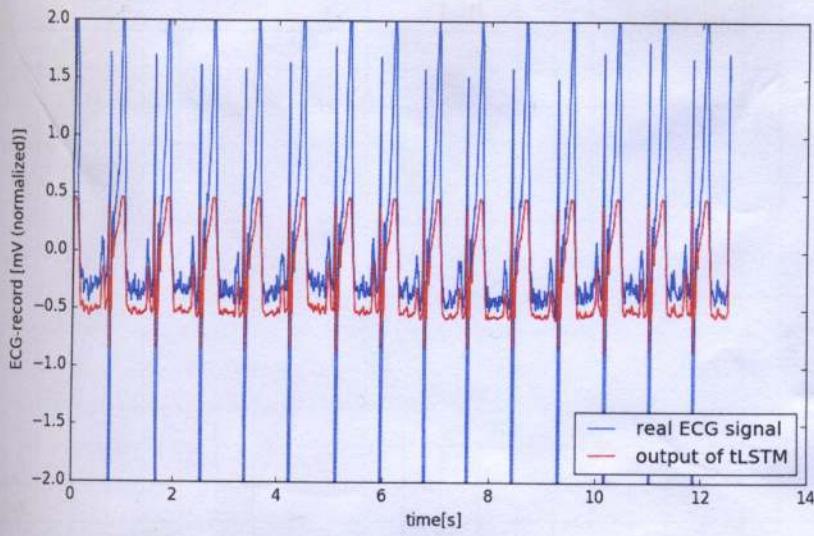
### PNC 2016: Erster Test der Konvertierung PCG $\rightarrow$ ECG



Erste Ergebnisse: Output sieht fast zu gut aus (nach 1. Epoche)

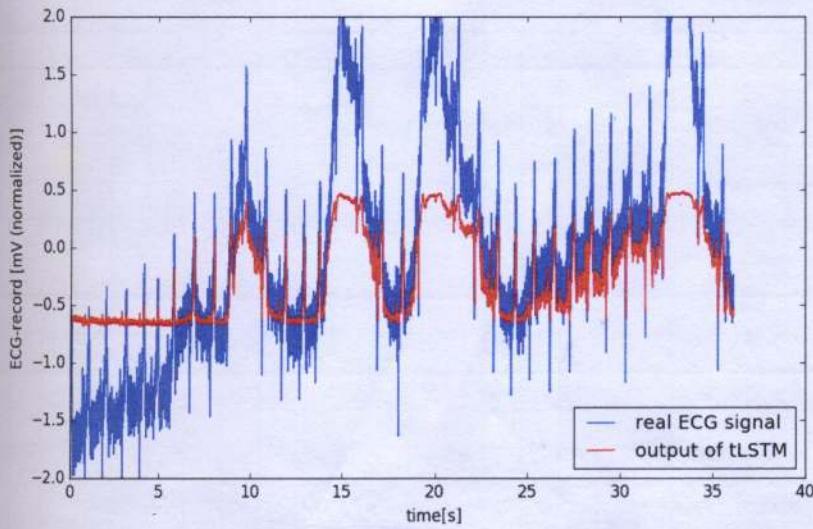
$\Rightarrow$  Fehler im Code? (Siehe Plots)

Problem: Training bringt keine Verbesserung!



data\_001.pug

results/pcg2ng-32hidden



data\_303.pug

Test-Suite: Algorithmus von Eckmann et al. für Lyapunov-Exp.

16.04.2016

Test-Suite enthält jetzt den Algorithmus von Eckmann et al.  
zur Approximation des Lyapunov-Exponenten.

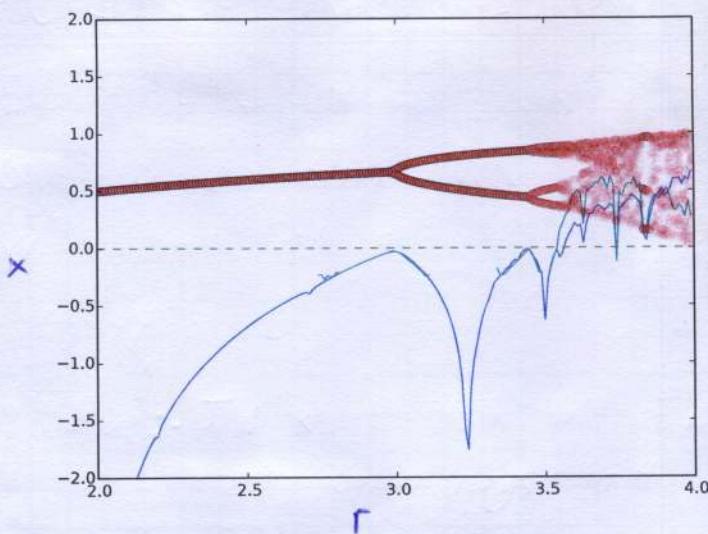
- Erste Beobachtungen:
- Sehr viele Positionen haben  $\exp -\infty$  bei logistic-map-Beispiel.
  - Wo der Ausgabewert nicht  $-\infty$  ist, ist er sehr nahe an dem analytisch berechneten "echten" Exponenten

Referenzen:

systems-scifues.uni-graz.at (analytische Lösung für Lyap-Exp  
der logistischen ~~Abbildung~~ Abbildung)

[mathworks.com](http://mathworks.com) → Lyapunov Exponent Toolbox  
R-Paket "tseriesChaos", Funktion lyap-k

Anmerkung: Vergleich mit kantz (lyap-k) nicht möglich, da die Parameterfindung für lyap-k schwer bis unmöglich ist.



Eckmann (fürkos)  
vs analytischer  
Lyap-Exponent (blau)  
rot: Bifurkationsplot

17.04.2016 PNC 2016: DTW als Ersatz für RMSE

dtw.py

Dynamic Time Warping sollte bessere Zielfunktion für  $\text{PCG} \rightarrow \text{ECG}$ -Training sein.

Erläuterung: Wichtig im ECG sind die (krassen) Peaks. Trifft man sie direkt, bekommt das Netz viele "Punkte", verfehlt es sie knapp, wird es doppelt bestraft. Einmal dafür, dass der Peak ~~fehlt~~ fehlt und einmal dafür, dass er an einer falschen Stelle auftaucht.

$\Rightarrow$  DTW würde dafür nur geringe Penalty geben.

DTW und erste Grundlagen für FastDTW in python implementiert.

Problem: DTW ist nicht ableitbar  $\Rightarrow$  Evolution training?

26.04.2016 Test-Suite: Hurst Exponent

hrv-nonlinear.py Nichtlineare algorithmen ausgelagert in hrv-nonlinear.py

Hurst Exponent mit rescaled range (R/S) implementiert.

Referenzen:

funktion "hurst" im R-Paket "pracma"

Matlab-Implementierungen auf mathworks.com und ideas.repec.org

Erste Beobachtungen: schwer zu testen ohne Daten mit klar definiertem Hurst-Exponenten.  
Wahl der Skalierungsstufen willkürlich?

PNC 2016: Einreichung des ersten Entrys

28.04.2016

Eingecktes Netz: Epoche 18 vom ersten Lauf von Ansatz ①

Ergebnis: Score: 0.71

sensitivity: 0.63

specificity: 0.79

PNC 2016: Test-klassifizierer auf Basis der künstl. ECGs

29.04.2016

Aufbau: LSTM mit 100 Einheiten im Hidden Layer  
 $\Rightarrow$  gleiche Netzstruktur wie in Ansatz ①

Ergebnis:  $\approx 50\%$  accuracy  $\Rightarrow$  Zufall

$\Rightarrow$  Klassifizierung macht erst mit Evolino-Netzen Sinn

Test-Suite: Correlation Dimension und DFA

Correlation Dimension und Detrended Fluctuation Analysis (DFA)  
 zu hrv-nonlinear.py hinzugefügt.

Referenzen (Correlation Dimension):

- "corrDim"-Funktion in R-Paket "fractal"
- mathworks.com, Peng Yuchuan, "Correlation Dimension"

Referenzen (DFA):

- C-Code von physionet.org
- Funktion "DFA" in R-Paket "fractal"
- "Introduction to MDFA in Python"  $\Rightarrow$  bsp.math.nikken.jp

Beobachtungen: Implementierungen scheinen für künstliche Minimal-Beispiele zu funktionieren, müssen aber noch für <sup>typische</sup> ~~reale~~ Probleme getestet werden.

30.04.2016

## PNC 2016: Implementierung des Evolino-Algorithmus

Evolino direkt mit numpy implementiert.  
 Gewichte von Keras-Netz werden als array ausgelernt,  
 im Algorithmus verändert und zur Evaluation wieder  
 eingefügt.

Probleme: - PTW klappt auch mit Evolino nicht, da der  
 letzte Optimierungsschritt linear ist.

06.05.2016

## PNC 2016: Erster Test des Evolino-Algorithmus

Ergebnis: Memory Error (and Absturz von Laden j))

Grund: Matrix für Least-squares wird zu groß.

Lösung: Iterative Berechnung der Least-Squares-Lösung

10.05.2016

## Test-Suite: Dokumentation fertiggestellt + Rosenstein-Algorithmus

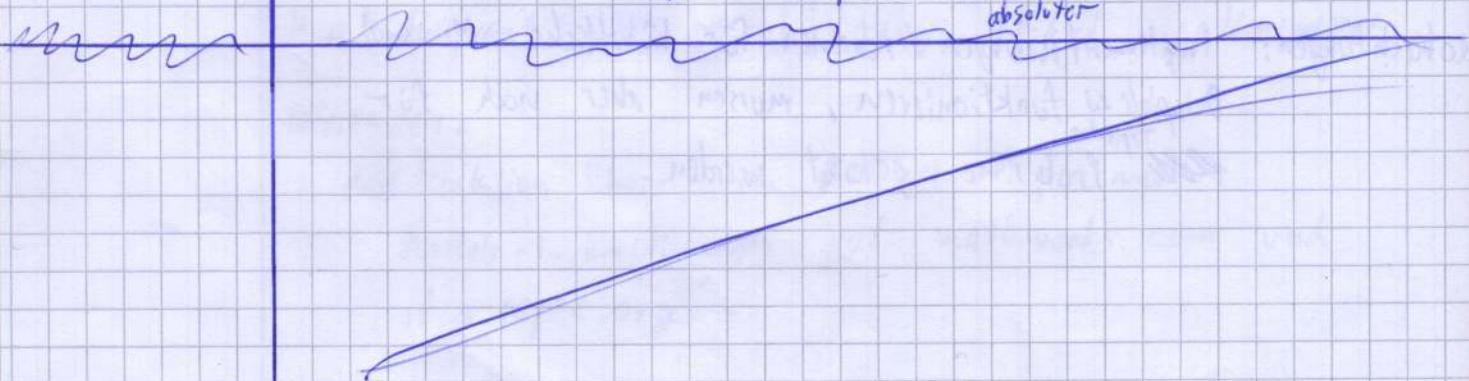
Umfangreiche Doku mit Ziel, dass der Code die  
 Algorithmen und die Maße, die sie erzeugen erklären kann

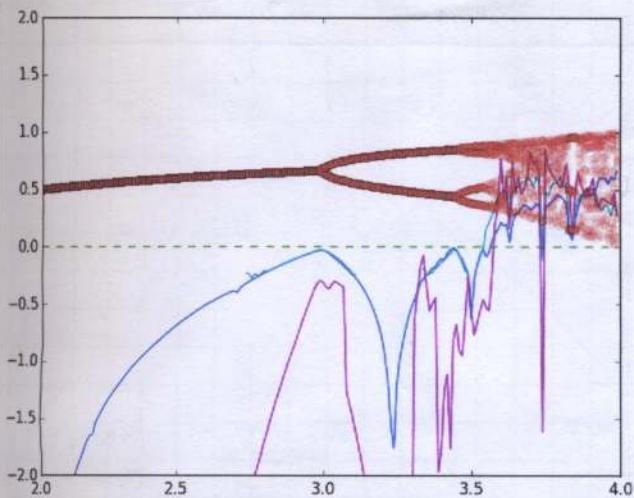
Außerdem: Implementierung des Algorithmus von Rosenstein et al.  
 zur Approximation des größten Lyapunov-Exponenten.  
 (Sollte für 1D-System robuster sein als Eckmann)

Referenzen:

[mathworks.com](http://mathworks.com), mirwais, "Largest Lyapunov Exponent."  
[ideas.repec.org](http://ideas.repec.org), Shapour Mohammadi, "Lyaprosen"

Beobachtung: Weniger Stellen mit  $-\infty$ , Vorzeichen stimmt  
 bei logistic map, aber Wert schwankt stark.





Violett: Rosenstein-Algorithmus

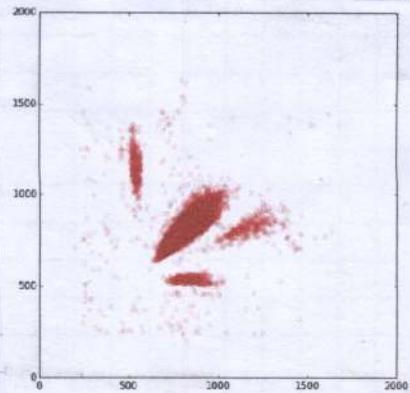
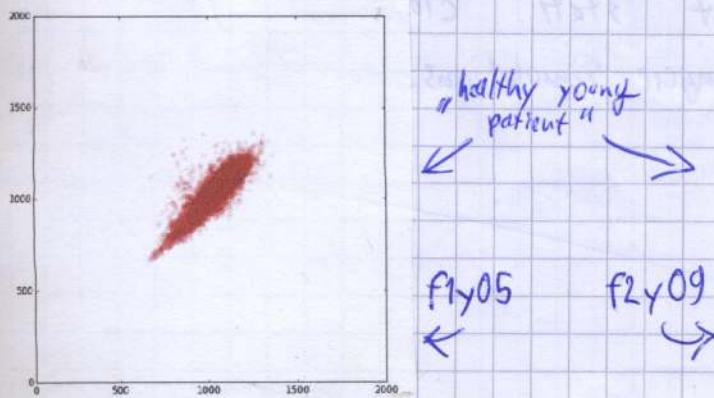
Test-Suite: Sammlung von menschlichen HRV-Daten zum Test der Algo. 24.05.2016

Physionet-Datenbank mit gesunden/"normalen" HRV-Daten:

- Fantasia - DB: labelled ECGs of healthy patients (young vs old)
- PTB diagnostic ECG database: 290 patienten inkl. Diagnose, 52 „healthy controls“, ECG
- Physiologic response to changes in posture: 10 healthy subjects, D:\Daten\physiost  
response to slow tilt, fast tilt and standing up, ECG
- 428 No. MIT-BIH normal sinus rhythm database: 18 long-term ECG referred to hospital but with normal sinus rhythm, D:\Daten\hrvdb
- Normal sinus rhythm RR-Interval database: 54 long-term ECGs, only RR-intervals in database, subjects have „normal sinus rhythm“ (not necessarily „healthy“)

RR-Intervalle (bzw. beat timestamps) extrahiert mit  
mdann, geparsed mit python, zusammengefasst in npz-Format.

Beobachtung: Sehr unterschiedliches Bild im Poincaré-Plot selbst  
innerhalb der healthy/normal subjects.



29.05.2016

## PNC 2016: Denoising Recherche zu Denoising mit Wavelets

HR

PCG  $\rightarrow$  ECG macht mit aktueller ECG-Qualität nicht viel

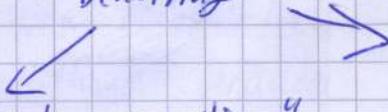
1.

Sinn  $\Rightarrow$  Denoising

„Baseline wander correction“



notch filter

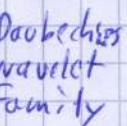


wavelet-based denoising

discrete wavelet transform

Daubechies wavelet family  $\xrightarrow{\text{more symmetry}}$ 

Symlet wavelet family

Vermischte Pipeline: Rohsignal  $\rightarrow$  notch filter  $\rightarrow$  Sym 12 dwt  
daub 5

06.06.2016

## HRVDB: Erkenntnisse aus der Literatur zu Poincaré-Formen

2.

Esperer et al. "Cardiac Arrhythmias Imprint Specific Signatures on Lorenz Plots" (2008)

- Identifiziert einige typische Poincaré-Figuren und verknüpft sie mit Krankheitsbildern

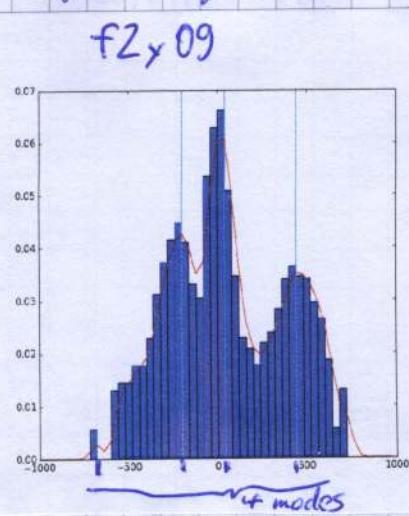
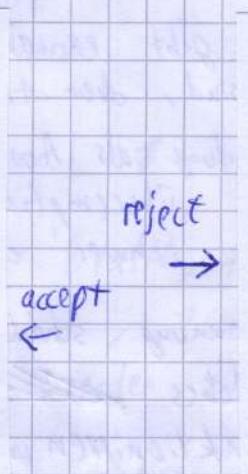
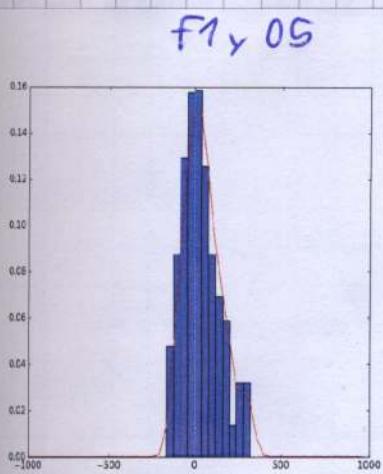
Gedanken dazu: Einige "healthy" samples in den Datenbanken zeigen "Fächer" oder "Double-sided-lobe pattern"  
 $\Rightarrow$  Möglicherweise doch keine normalen Sinusrhythmen?

$\Rightarrow$  Automatische Beat-Extraktion nochmal überprüfen

- Macht es Sinn, nur Torpedo- u. komplexen Formen als Referenz zu verwenden?
- Wie ändert sich die Form, wenn man nur 100s-Samples betrachtet?
- Side-Note: Kann man auf Basis dieser Daten ein LSTM trainieren, das Poincaré-Formen klassifiziert? (Vielleicht sogar mit fuzzy Klassen)  
 $\Rightarrow$  "fuzziness" = 0.7 statt class = "fun"
- Muss man vielf. längere Simulationszeiten für Test-Suite nehmen? ~~etwas~~

# HRVDB: Metriken zur Erkennung von krankhaften Poincaré-Patterns 09.06.2016

1. Idee: Projektion aller Punkte im Poincaré-Plot auf Achse  $(-1, 1)$   $\Rightarrow$  Histogramm  $\Rightarrow$  uni-/multimodal?  
 $\Rightarrow$  gauss smoothing  $\Rightarrow$

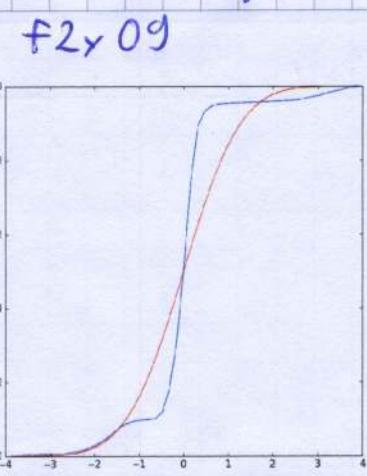
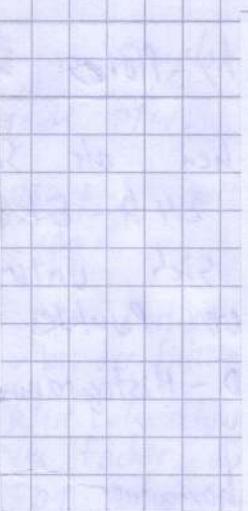
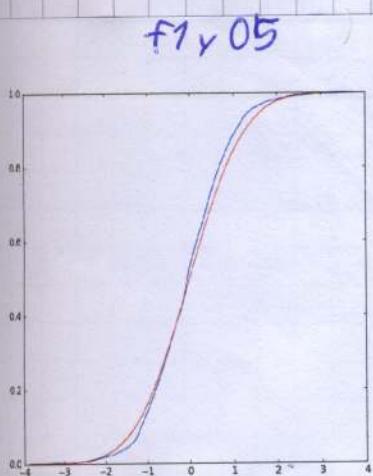


hrvdb\plots...

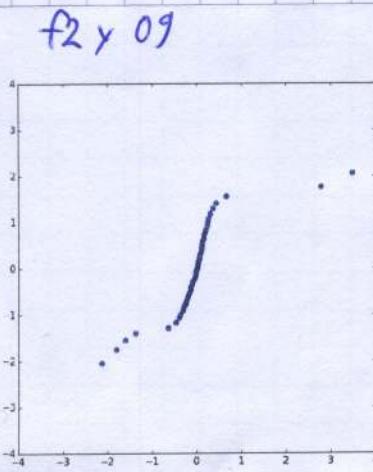
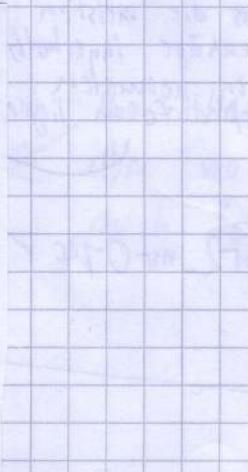
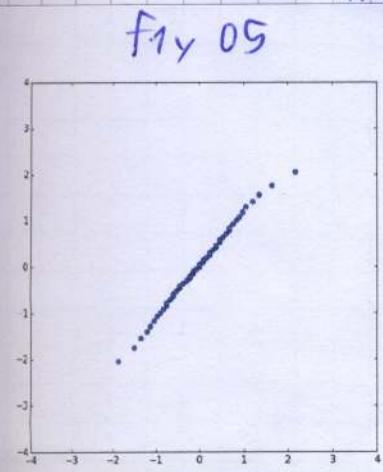
2. Idee: Shapiro-Wilk / anderr Test für Normalverteilung.

Frage: Sind Punkte entlang  $(-1, 1)$  normalverteilt?

$\Rightarrow$  Plot normal CDF vs. cumsom von Histogramm.



Zusätzlich: QQ-Plot ~~von~~ Normalverteilung approximierte Quantile der ~~Pattern~~ Histogramm.



Beobachtung: Die jungen Patienten der Fantasia-Datenbank (Healthy young) folgen sehr gut der Normalverteilung, aber es gibt insges. sehr viele Datensätze, die zwar normal aussehen, aber nicht ~~normalverteilt~~ sind.

Anderer herum gibt es aber auch keine Datensätze, die abnormal sind, aber trotzdem normalverteilt.

⇒ Normalverteilung als Ausschlusskriterium funktioniert, ist aber "überempfindlich" (und außerdem ohne Q-Q-Plot schwer zu quantifizieren)

Multimode-Erkennung schont dagegen für double/triple side-lobes und island patterns gut zu funktionieren und schlägt nur für fan patterns fehl.

⇒ Einfachste Erkennung: Multimode + Varianz

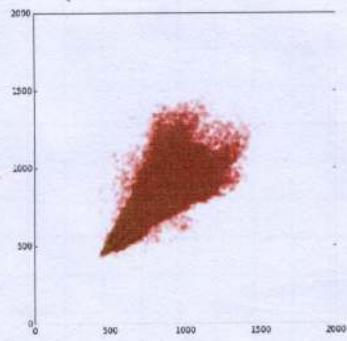
(Raudortiz: Erkennung der visuellen Patterns mit LSTM wäre wirklich lohnend für ein Paper, falls noch nicht existent)

10.06.2016

HRVDB: Poincare als density-plots

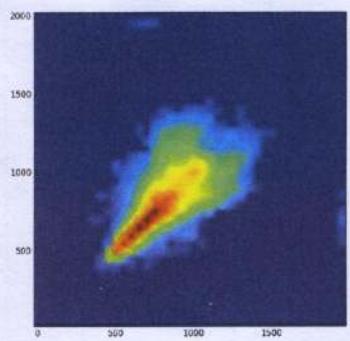
Idee: Poincare-Plots machen als Scatterplots eigentlich wenig Sinn. Bei 24 h-EKG kann man oft nicht erkennen ob sich unter einem Punkt 10 oder 100 andere Punkte verborgen.

⇒ Density-Plot (2D-Histogramm)



↙ homogener  
„Fächer“

Dichteplot zeigt,  
dass die meisten  
Herzschläge innerhalb  
einer normalen  
Torpedoform liegen



↖ ↗  
msr2-msr074

# HRVDB: Erste Implementierung eines Filter-Schemas

Histogramm (log):

- 50 bins im Intervall  $[-1000, 1000]$
- gauss-kernel Länge 23,  $\sigma = 0.8$
- maxima akzeptiert falls  $> 0.01$
- normales histogramm  $\rightarrow \log \rightarrow \text{norm}$

SD1:

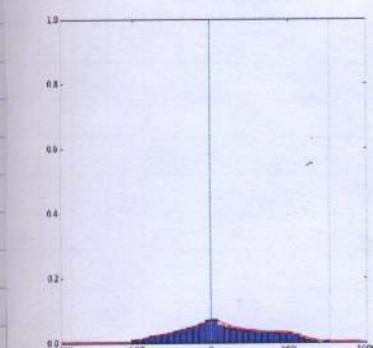
- projektion auf  $(-1, 1)$
- berechnung der Stdabw.
- cutoff: 70  
(mean  $SD1 = 42 \pm 7$   
laut Guzik 2007)

Filterbedingung:  $|\text{local max.}| = 1 \wedge SD1 < 70$

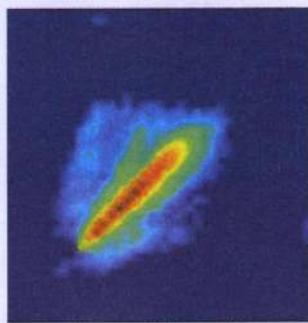
Ergebnis (Anwendung auf 'healthy\*'):

- 99 samples ausgeschlossen, 103 behalten
- Triplets/double side-lobe werden zuverlässig per Histogramm erkannt
- Breite Streuung im Plot führt aber nicht immer zu hohem  $SD1 \Rightarrow$  false ~~positives~~ positives mit „Fächer“
- Idee: Bei Fächermuster ist oft die Höhe des Absoluten Maximum im Histogramm sehr klein.  
 $\Rightarrow$  Cutoff anhand Verteilung der Daten um Mittelpunkt?  
( $\times\%$  müssen innerhalb  $[n - \sigma, n + \sigma]$  liegen)
- bei einigen ausgeschlossenen samples scheinen multimode- oder  $SD1$ - Kriterium noch etwas hart
- bei kurzen samples aus ptb wirkt  $SD1$ - Kriterium ebenfalls nur unzureichend.

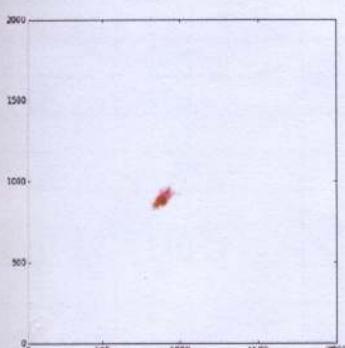
hrvdb/filter



false positive  
bei der Filterung  
klare Entwicklung  
zum Fächer, obwohl  
 $SD1$  ist „normal“



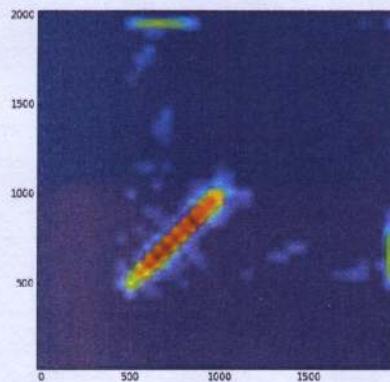
nsr2-nsf017



Vom Filter akzeptiert, sieht  
aber noch genauer Betrachtung  
auch fächerartig aus

Zusätzliche Beobachtung:

Density-Plot zeigt teilweise extreme outlier im Poincaré-Pot, vor allem bei nsr2.  
( $\Rightarrow$  SD7 wird riesig)

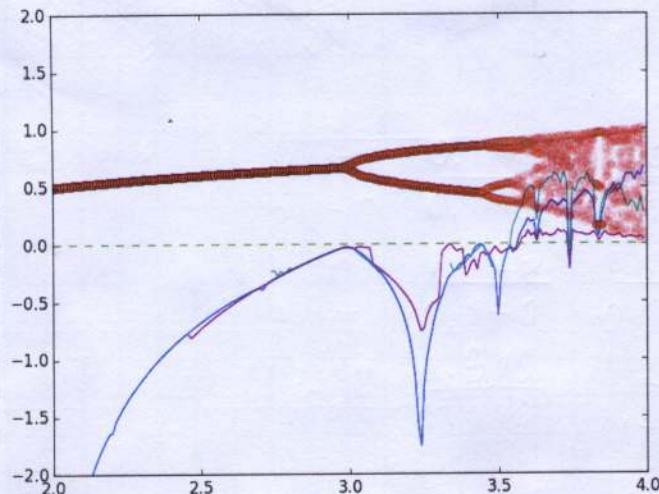


density plot von  
nsr2\_nsr016

16.6.2016 ~~Bugfix~~ Bugfix für Rosenstein - Algorithmus

Rosenstein - Algorithmus zum Berechnen des größten Lyapunov - Exponenten hatte unötigerweise auch den Index (x-Achse) der Trajektorie logarithmisch.

Sowohl Florian Görg als auch Internet - Varianten (mirwals, Shapour Mohammadi) nutzen unlogarithmierte Indices, Paper von Rosenstein ~~sag~~ stimmt damit überein  $\Rightarrow$  Code gefixed. Neue plots s.u.:



Blau: analytischer MLE  
Türkis: Eckmann  
Röd: Rosenstein (fixed)

results/chaos/  
lyap\_r-fixed.png

Außerdem weiterer kleiner Fix: Daten werden nach float 32 konvertiert (outlier führen zu integer - Overflow in euler-Distanz)

Outlier in NSR2: Datensatz: nsr2\_nsr034  
~~Herzschlag-Länge~~  
 Herzschlag-Index: 67  
 Herzschlag Länge: 28659

Bugfix für Hurst mit R/S - Analyse

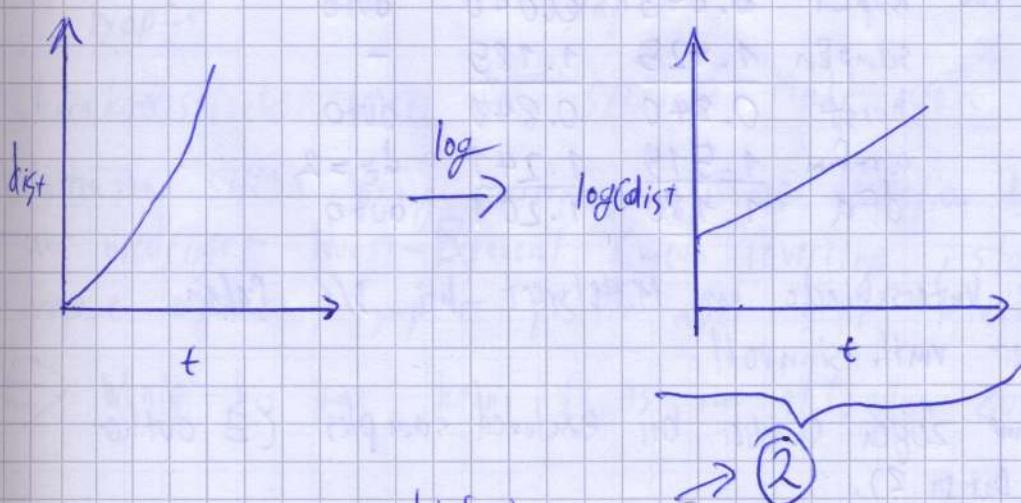
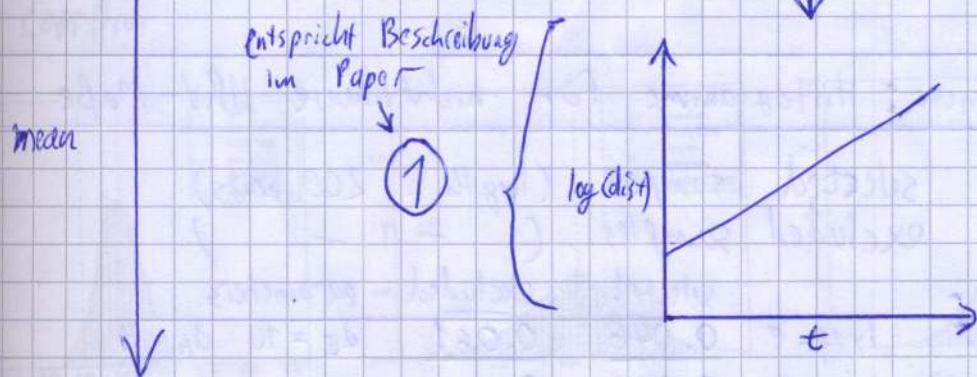
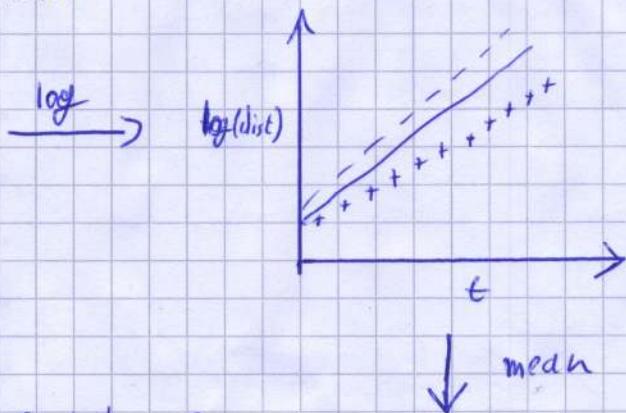
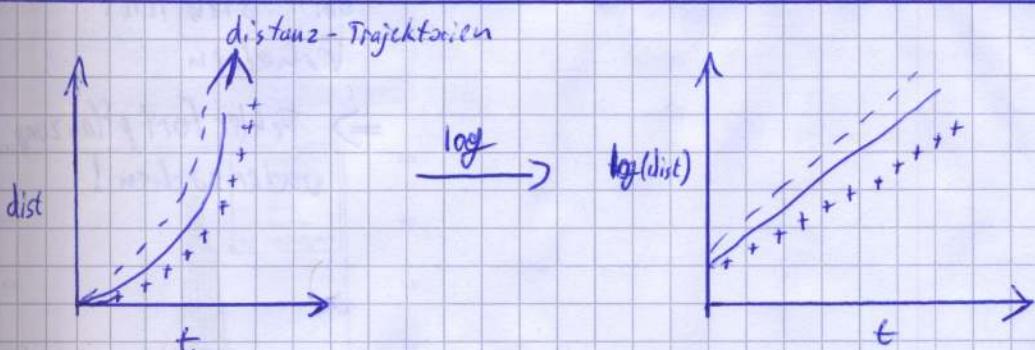
17.06.2016

Wenn lange Strecken im Signal den gleichen Wert haben kann  $R=0$  werden  $\Rightarrow S=0 \Rightarrow R/S = NaN$

Lösung: NaNs aus mean und Line-fitting herausnehmen  
 Falls alle ~~NaNs~~ <sup>(R/S)\_n</sup> NaNs sind  $\Rightarrow$  Ergebnis = NaN

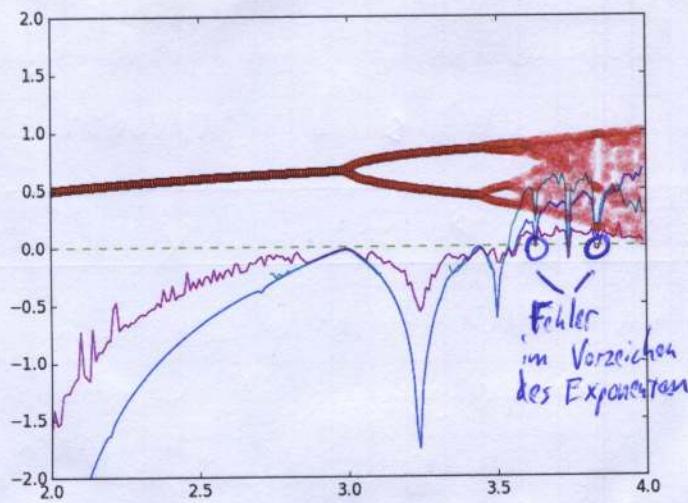
Bugfix für Roseinstein - Algorithmus: Mean(Log) statt Log(Mean)

20.06.2016



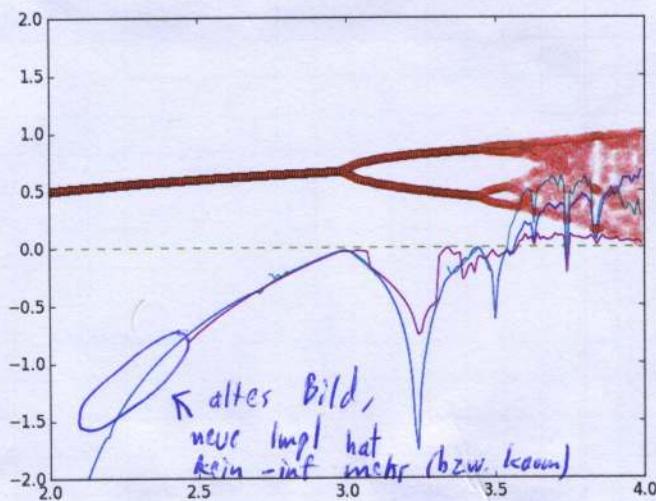
①

results/chaos/  
lyap\_r\_fixed-  
mean-of-log.pug



Eigentlich korrekt  
Variante, sieht  
aber (ggf bei welcher  
Parameterwahl)  
„instabiler“ aus.

②



Eigentlich falsch,  
aber stabiles  
Verhalten  
⇒ Fehlerfortpflanzung  
untersuchen!

21.06.2026 HRV-Nonlinearity: Histogramme für nichtlineare HRV-Maße

8586	selected samples	(length : 200 beats)	
30605	excluded samples	( - 11 - )	
	selected	excluded	parameters
Ergebnis:	Tyap-e	0.046	$d_E = 10$ $d_H = 4$
	lyap-r	0.043	auto
	sampEn	1.425	1.185
	hurst	0.840	0.849
	corrDim	1.519	1.241
	dFa	1.180	1.201

Merkliche Unterschiede im Mittelwert bei 3/6 Maßen  
⇒ Filter vmtl. sinnvoll

Histogramme zeigen Outlier bei excluded samples ( $\exists$  outlier  
in den Daten?).

## Unterschiedliche Verteilungsmuster:

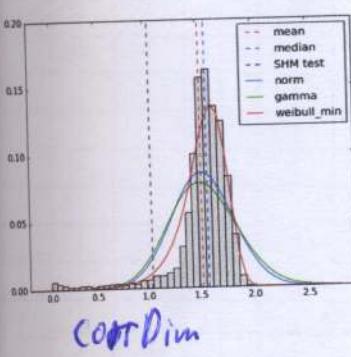
corrDim, dfa undhurst scheinen gamma-verteilt zu sein  
lyap-e, lyap-r und sampEn dagegen Cauchyverteilt

## Histogramme mit PDFs unterschiedlicher Verteilungen 26.06.2016

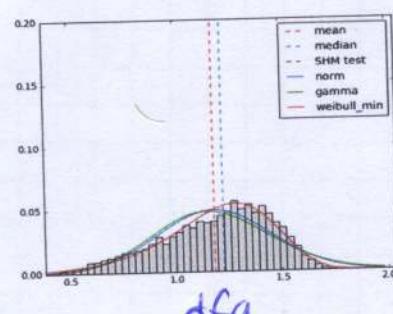
Für left-skewed Verteilung gibt es drei "typische" Kandidaten, die auch in scipy vertreten sind:

- beta: sehr flexibel (fittet quasi alles)
- gen. logistic: gute Abdeckung, hohe Maxima
- Weibull: verwendet für "mean time to failure"-Kalkulationen, kann neg. u. pos. skew haben, bester fit.

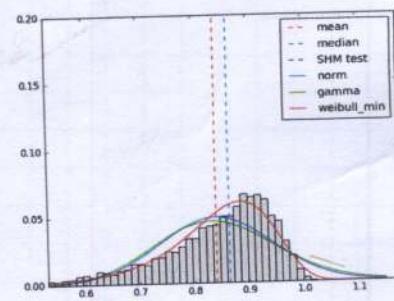
Promotion / results  
/chaos / nonlinear  
-measures\_1.0



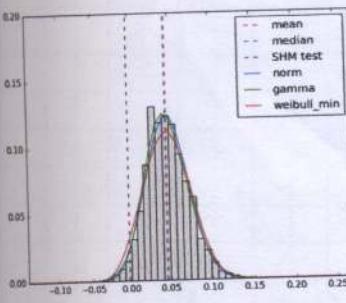
corrDim



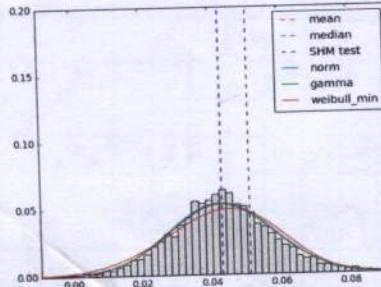
dfa



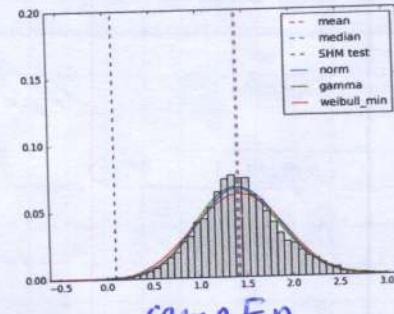
hurst



lyap-e



lyap-r



sampEn

Charakteristischer Wert für SHM: Mode von Weibull?

Verhalten von SHM: Wenig Entropie, geringere Correlation dimension, sehr niedriger Hurst-Exponent (mean-reverting, stationary), lyap-e negativ, lyap-r positiv aber sehr klein

⇒ Wenig bis gar kein Chaos im aktuellen Zustand.

26.06.2016

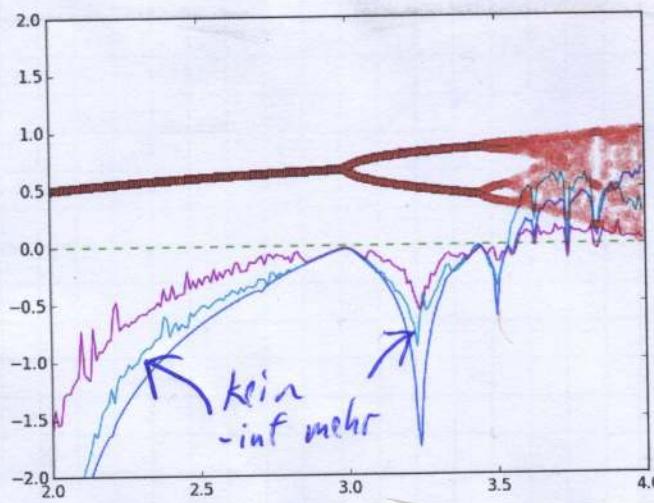
Idee: RANSAC für Line-fitting?

Manche Plots haben auch bei selected samples outlier  
 $\Rightarrow$  robustere Line-fitting-Methode wie z.B. RANSAC

27.06.2016 Zero-Safety Update: Keine -inf-Werte mehr in hrv.nl

Wo möglich werden Nullen vor dem Logarithmieren gefiltert  $\Rightarrow$  weniger Datenpunkte, aber -infinity wird vermieden.

Große Änderung bei lyap-e:

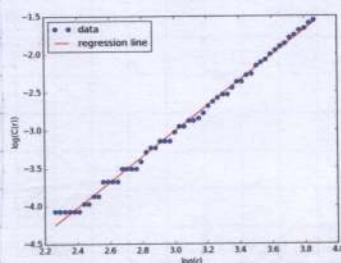


lyap-r und lyap-e sehen sich jetzt ähnlicher.

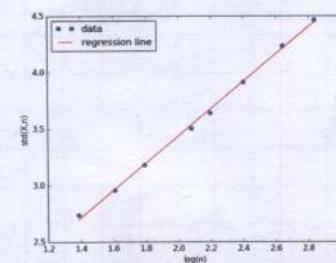
HRV-nonlineair mit multiprocessing auf Zadon

Thread pool ~~gezogen~~ mit  $\checkmark$  Patenbank-Samples als Input.  $\Rightarrow$  Prozess funktioniert für kleine Datenmengen, für große stoppt die Ausführung nach abarbeitung der selected-Patenbank scheinbar.

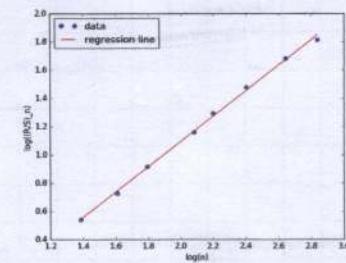
HRV-nl: Plots für alle nichtlinearen Maße



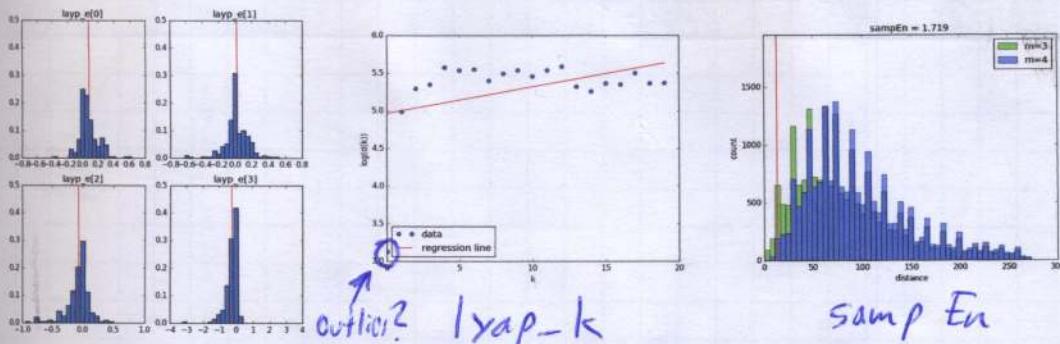
corr Dim



dfa



hurst



lyap-e

## HRV-n1: Multiprocessing Fix

08.07.2016

Nach langem Debuggen ist der Fehler gefunden, der einen oder zwei Worker-Prozesse in Deadlock wirft (in dem sie trotzdem 100% CPU-Leistung verbrauchen):

Details:  
Notizen / allgemein.txt

OpenBLAS (C-library für Matrixoperationen) ist multi-threaded, kann aber nicht mit multi-threading in der aufrufenden Anwendung umgehen

⇒ Wenn OpenBLAS mit multi-threaded Anwendung verwendet wird muss OPENBLAS\_NUM\_THREADS (UmgangsvARIABLE) auf 1 gesetzt werden.

## PNC: Training von LSTM auf rohen EKG-Daten

11.07.2016

Name für Paper: LSTM-e

Gleiche Netzstruktur wie Ansatz ① (LSTMp)

1 Hidden-Layer mit 100 LSTM-~~Neuronen~~ Units

Datm. training-a/\* (Alle normalen (105) + 105 zufällig ausgew. abnormalen)

Forschung/2016/thomas-pnc2016/data/results/lstm-e-100hidden

Teilung Training/Test: 90% / 10%

Rheinfeld: Abwechselnd normal / abnormal

Epoch 11

Training		Test	
gt-a	gt-n	gt-a	gt-n
ca 65	22	ca 8	3
ca 41	83	ca 3	9

Accuracy Train LSTMp: 82,31%

Accuracy Train: 70,14 %

Accuracy Test: 73,91 %

⇒ Nur 12,17 % Accuracy-verlust ⇒ EKG enthält auch relevante Informationen

17597907 function calls in 2312.657 seconds

ncalls	tottime	percall	cumtime	percall	filename:lineno(function)
99989	0.259	0.000	0.345	0.000	<frozen importlib._bootstrap>:2264(_handle_fromlist)
1	0.001	0.001	2312.657	2312.657	<string>:1(<module>)
99988	0.559	0.000	620.409	0.006	_methods.py:25(_amax)
99988	0.103	0.000	1.134	0.000	_methods.py:31(_sum)
199976	0.284	0.000	0.992	0.000	fromnumeric.py:1291(ravel)
99988	0.482	0.000	1.806	0.000	fromnumeric.py:1631(sum)
99988	1.471	0.000	621.879	0.006	fromnumeric.py:2058(_amax)
99988	0.659	0.000	587.745	0.006	fromnumeric.py:803(argsort)
199976	0.204	0.000	0.204	0.000	getlimits.py:255(min)
199976	0.227	0.000	0.227	0.000	getlimits.py:269(max)
1	1052.254	1052.254	2312.656	2312.656	hrv_nonlinear.py:209(lyap_e)
1	0.038	0.038	0.038	0.038	hrv_nonlinear.py:300(<listcomp>)
99988	2.169	0.000	2.169	0.000	hrv_nonlinear.py:348(<listcomp>)
299964	0.635	0.000	1.868	0.000	linalg.py:106(_makearray)
699916	0.352	0.000	0.664	0.000	linalg.py:111(isComplexType)
399952	0.258	0.000	0.396	0.000	linalg.py:124(_realType)
99988	0.027	0.000	0.027	0.000	linalg.py:130(_linalgRealType)
199976	1.023	0.000	1.879	0.000	linalg.py:139(_commonType)
199976	0.523	0.000	0.629	0.000	linalg.py:168(_to_native_byte_order)
99988	6.434	0.000	23.508	0.000	linalg.py:1733(lstsq)
399952	0.864	0.000	1.796	0.000	linalg.py:180(_fastCopyAndTranspose)
199976	0.340	0.000	0.370	0.000	linalg.py:192(_assertRank2)
99988	0.107	0.000	0.107	0.000	linalg.py:219(_assertNotEmpty2d)
99988	3.151	0.000	13.452	0.000	linalg.py:607(qr)
99989	0.690	0.000	1.936	0.000	numeric.py:2125(identity)
799904	0.976	0.000	2.147	0.000	numeric.py:394(asarray)
99988	0.126	0.000	0.317	0.000	numeric.py:464(asanyarray)
99989	0.721	0.000	0.902	0.000	twodim_base.py:190(eye)
199976	0.490	0.000	0.921	0.000	twodim_base.py:22(_min_int)
299964	1.036	0.000	2.019	0.000	twodim_base.py:242(diag)
99988	0.624	0.000	3.425	0.000	twodim_base.py:372(tri)
99988	0.670	0.000	5.155	0.000	twodim_base.py:459(triu)
499940	0.883	0.000	0.883	0.000	(built-in method _fastCopyAndTranspose)
99988	0.030	0.000	0.030	0.000	(built-in method abs)
199976	0.776	0.000	0.776	0.000	(built-in method arange)
1299845	4.042	0.000	4.042	0.000	(built-in method array)
199976	6.775	0.000	6.775	0.000	(built-in method dgelsd)
199976	0.990	0.000	0.990	0.000	(built-in method dgeqr)
199976	0.776	0.000	0.776	0.000	(built-in method dorgqr)
299964	1.315	0.000	1.315	0.000	(built-in method dot)
99988	0.115	0.000	0.115	0.000	(built-in method empty)
1	0.000	0.000	2312.657	2312.657	(built-in method exec)
299964	0.158	0.000	0.158	0.000	(built-in method getattr)
199976	0.086	0.000	0.086	0.000	(built-in method hasattr)
99988	0.190	0.000	0.190	0.000	(built-in method isinstance)
999880	0.574	0.000	0.574	0.000	(built-in method issubclass)
1599810	0.301	0.000	0.301	0.000	(built-in method len)
99988	0.184	0.000	0.184	0.000	(built-in method log)
299964	0.770	0.000	0.770	0.000	(built-in method max)
699917	0.316	0.000	0.316	0.000	(built-in method min)
499942	3.581	0.000	3.581	0.000	(built-in method where)
1499823	2.680	0.000	2.680	0.000	(built-in method zeros)
399952	0.053	0.000	0.053	0.000	(method '_array_prepare_' of 'numpy.ndarray' objects)
299964	0.083	0.000	0.083	0.000	(method 'append' of 'list' objects)
99988	587.086	0.006	587.086	0.006	(method 'argsort' of 'numpy.ndarray' objects)
199976	0.334	0.000	0.334	0.000	(method 'astype' of 'numpy.ndarray' objects)
199976	0.440	0.000	0.440	0.000	(method 'copy' of 'numpy.ndarray' objects)
199976	0.178	0.000	0.178	0.000	(method 'diagonal' of 'numpy.ndarray' objects)
1	0.000	0.000	0.000	0.000	(method 'disable' of '_isprof.Profiler' objects)
399952	0.137	0.000	0.137	0.000	(method 'get' of 'dict' objects)
99988	0.901	0.000	0.901	0.000	(method 'outer' of 'numpy.ufunc' objects)
199976	0.266	0.000	0.266	0.000	(method 'ravel' of 'numpy.ndarray' objects)
199976	620.881	0.003	620.881	0.003	(method 'reduce' of 'numpy.ufunc' objects)

Profiling eines Aufrufs von lyap-e mit Sequenzlänge 100.000.

Beobachtungen:

- Groß die Hälfte der Zeit ~~benötigt~~ benötigt für numpy-calls, die vmtl. nicht umgehbar sind.
- Andere Hälfte vmtl. für Loops/List-comprehensions in Hauptfunktion benötigt.

⇒ Maximal zu erwartender Speedup = 2x

⇒ Macht vmtl. mehr Sinn den Algorithmus selbst zu optimieren (Anzahl betrachteter Punkte begrenzen?)

## PNC: Refactoring des Evolino-Codes für EvoDTW

25.07.2016

~~Evo~~ Abstrakte Basisklasse für Evolution,

- Evolino : LSTM mit Evolution, Output mit LLSQ, Objective : MSE
- EvoDTW : LSTM + Output mit Evolution trainiert, Objective : DTW

Erster Test (Lernen der kumulativen Summe):

- EvoDTW lernt sehr langsam
  - Erst nach ca. 20 Epochen mit 1000 Auswertungen pro Epoche zeichnet sich "Lernfortsch" ab.
- ⇒ Wenn DTW, dann vlt. doch lieber direkt in Theano implementiert mit Scan.

## PNC: Manuelles Aussortieren der EKGs in training-a

07.08.2016

Jedes EKG von Hand angesehen und ~~mit~~ mit Tags versehen:

- nice : klarer PQRST-Komplex ~~zu~~
- semi-nice : P, QRS und T klar zu erkennen, aber "untypisch"
- broken/strange : Signal extrem verschoben / nicht als EKG erkennbar
- upside-down : Vorzeichen vertauscht durch vertauschte Elektroden
- Notizen : power-line-interference und andere interessante Muster

⇒ Nur 721 von 405 Datensätzen wirklich brauchbar (nice)

## PNC: Korrektur zu confusion matrices

02.08.2016

Labeling der Confusion Matrices war verwirrt wg.

kontra-intuitiver Labels in PNC ( $1 = \text{abnormal}$ ,  $-1 = \text{normal}$ )

⇒ Eigenl. Layout aller bisher generierten F-Matrizen :

	true normal	true abnormal
predicted normal		
predicted abnormal		

02.08.2016

PNC: Denoising von PCG und ECG in training-a PN

- Idee aus Literatur: Butterworth high-pass filter für BW-removal H  
 => Verhält sich seltsam (fügt teilw. Baseline-Wander hinz) v  
 u. ist eigentlich zu kompliziert A  
 => stattdessen einfaches ~~BSpline Filter~~ Abschneiden HR  
 von Fourier-Koeffizienten mit weichen Kanten NC  
 (sigmoid) Git  
 Final Auswahl für Denoising:

ECG:

- Fourier- "Filter" mit cutoff bei 0.3 Hz für baseline Py  
 wander removal Vers
- NeighBlock wavelet - denoising mit sym12 u.  $\sigma = 0.08$  PN

PCG:

- NeighBlock wavelet - denoising mit sym12 u.  $\sigma = 200$  L

$\Rightarrow$  2016-thomae-pnc2016 / data-denoised

Zusätzlich Auswahl von "nice" samples

$\Rightarrow$  2016-thomae-pnc2016 / data-denoised-nice

Mögliche Verbesserungen (nicht implementiert):

- power-line interference entfernen
- $\sigma$  für jedes sample dynamisch bestimmen
- EMD statt wavelet

02a

PNC: Experimente mit LSTM u. CNN (pcg2ecg)

LSTM-Netze lernen nichts, auch auf denoised-nice  
 transformations-  $\nwarrow$  32 hidden units (1 layer)

$\Rightarrow$  Versuch mit CNN zu arbeiten

- CNNs brauchen BLAS-library  $\Rightarrow$  OpenBLAS installiert optimiert
- download OpenBLAS binaries in Ordner C:\OpenBLAS
- DLLs müssen in PATH sein (C:\OpenBLAS\bin)
- Theano-Flags: -LC:\OpenBLAS\bin -lopenblas
- leider führen alle Versuche nach 1. epoch zu NaNs

2016-thomae-pnc2016  
 ChSchoel\pcg2ecg  
 -cnn-test

PNC: Rcg2pcg statt pcg2rcg

Herzgeräusch kommt nach Depolarisation  $\Rightarrow$  funktioniert  
umgekehrter Weg besser?

Antwort: Nein.

HRV-nl: Chaosalgorithmen als offizielles Python-Paket

03.08.2016

Name: NOnlinear measures for Dynamical Systems (nolds)

Github: <https://github.com/CSchoel/nolds>

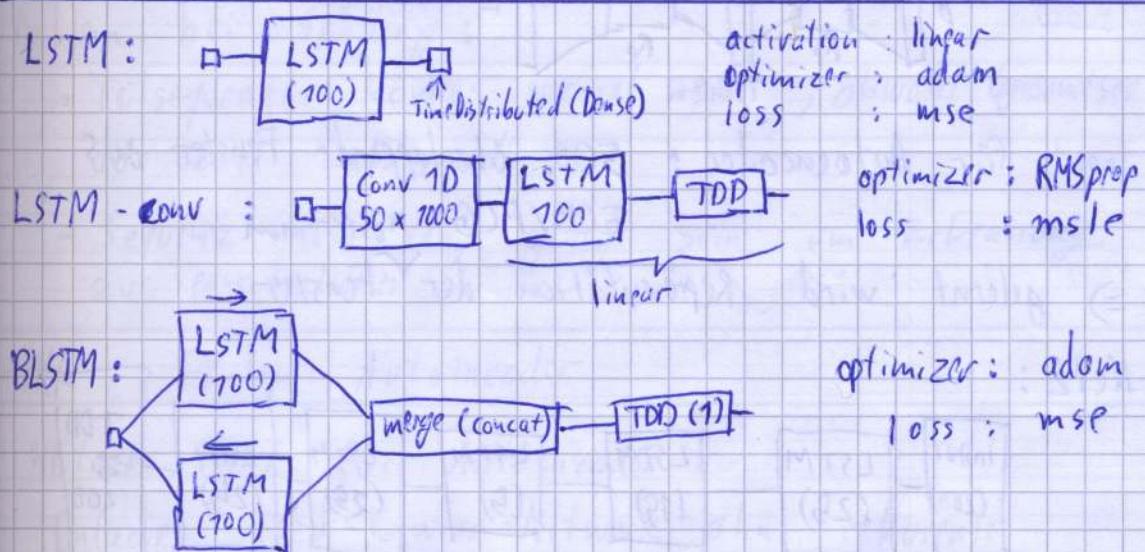
Promotional code!  
nolds

PyPi: pip install nolds

Version: 0.7.0

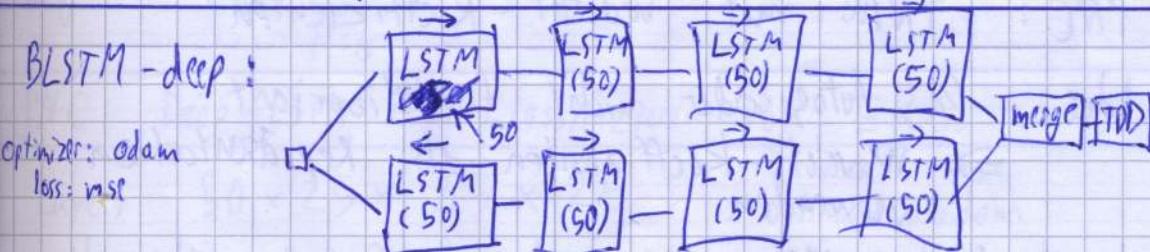
PNC: Verschiedene Netzstrukturen für TNN (pcg2rcg)

04.08.2016



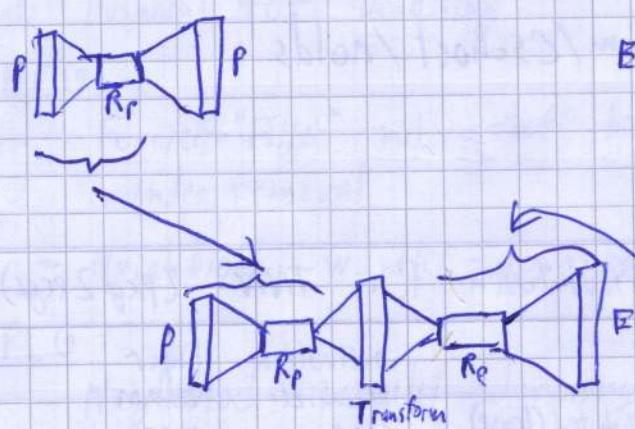
Ergebnis: Keins der Netze lernt etwas sinnvolles, BLSTM scheint aber die S-Zacke zu erkennen.

PNC: BLSTM-deep

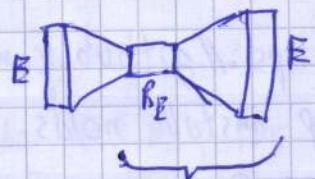


Idee: ~~PCG~~ Repräsentation, die ein Autoencoder lernt ist vllt leichter zu übersetzen, als die Daten selbst, da ein ECG-Decoder schon in der Lage sein müsste "fertige" QRS-Komplexe zu generieren.

PCG-Autoencoder

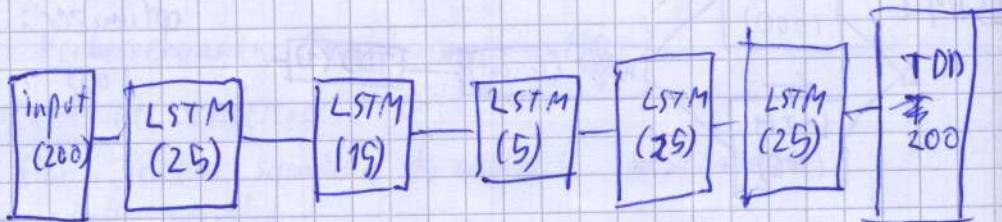


ECG-Autoencoder



Input für Autoencoder: 50% überlappende Fenster aus ECG/PCG Master im  
⇒ gelernt wird Repräsentation der ✓ Fenster

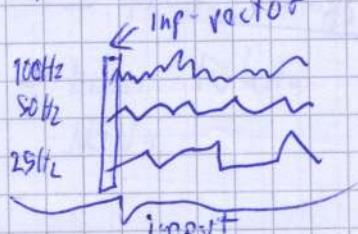
Netz:



Ergebnis: Netz lernt nicht einmal Identitätsfunktion, wenn LSTM-layer vergrößert werden. ☺

Idee: Da Autoencoder nicht funktioniert.  
⇒ Wavelet-Koeffizienten als Repräsentation verwenden

Input für TNN: Vektor der Koeffizienten über die Zeit verteilt.



Ergebnis: Leider kein sinnvoller Output erkennbar.

## PNC: split / mask input preparation

Zwei neue Arten, Input auf Training mit net.fit vorzubereiten:  
Batch-

- split: Teilt Datensatz in nicht überlappende Fenster fester Länge.
- mask: Fügt Sequenzen am Ende bis zu der maximalen Sequenzlänge mit einem Mask-value (0, NaN,  $\infty$ ,  $-\infty$ , ...) auf.

Problem beim Masking: ~~Nur der erste Layer~~

- Mask-Value kann nur beim ersten Layer ignoriert werden.

Problem bei Splitting:

- Teilsequenzen können normal sein, obwohl Gesamtseq. das Label "abnormal" hat.
- Sequenz muss lang genug sein um Erkennung zu ermöglichen.

## PNC: Fixlen Autoencoder

Weiterer Test für Autoencoder:

Einfaches Feed-Forward-Network ohne rekursive Verbindungen, angewendet auf gesplittete Sequenzen fester Länge.

Ergebnis: ~~Keine Änderung~~ zu bisherigen Autoencoder-Tests

## PNC: Deep-LSTM zum Klassifizieren von EKGs

Layers:  $50 \times 25 \times 25 \times 25$  loss: mse  
optimizer: adam

Data preparation: O-Masking, normalization to median 0 and std 1

Output: 2 Neuronen für normal/abnormal

Erste Test: Schlechter als ursprüngl LSTM 0.0

11.08.2016 PNC: Weitere Experimente mit ecg-classifier-deep

- ~~Kos~~ Split statt Masking
- weight regularization + Dropout  $\Rightarrow$  Reg. in LSTM führt zu NaN
- activation function:
  - softsign
  - ReLU  $\Rightarrow$  NaNs
  - linear  $\Rightarrow$  NaNs
- clipvalue in Adam to combat exploding gradients

$\Rightarrow$  Keine signifikante Verbesserung

13.08.2016

PNC: Erklärung für NaNs

NANs in einigen bisherigen Versuchen kommen von activation = "linear" in LSTM-layern.  
tanh ist eigtl. Standard-Aktivierungsfunktion für LSTM.

14.08.2016

PNC: Experimente mit EKG + PKG als Input

Ergebnis vergleichbar mit nur EKG als Input,  
 PKG-Klassifizierung mit neuen Netzen ist schlechter  
 als EKG-Klassifizierung . o.o

PNC: RplusLSTM-deep

Weitere Netzwerkstruktur:  $100 \times 100 \times 75 \times 75$  LSTM mit tanh-Aktivierung

$\Rightarrow$  NaNs egal bei welcher Einstellung (regularizer, cutoffs, ...)

18.08.2016

PNC: Großer Test für Klassifikationsparameter

Basis-einstellungen:

batch\_size = 30

window\_size = 10 000

net\_type = "new"

optimizer = "adagrad"

Test mit:

5

30 000

"old" (shallow LSTM)

"rmsprop"

Für jede Einstellung (Basis + jeder Test) 3 Netze

trainiert jeweils mit nur EKG, nur PKG und EKG+PKG als input. (jeweils nur aus training-a)

testfall	pcg		ecg		both	
	train	test	train	test	train	test
baseline	54	55	62	67	70	67
batch-size=5	51	55	51	51	56	57
rmsprop	69	69	58	61	70	70
old net	69	44	63	49	83	46
window-size=30k	82	63	65	72	78	80

Accuracy [%]

data/results/  
sched

⇒ Größere Fensterbreite und RMSprop scheinen zu helfen.

PNC: Test mit rmsprop u. window-size 30.000

19.08.2016

ws = 30k, rmsprop	pcg		ecg		both	
	train	test	train	test	train	test
	92	89	62	62	96	78

- Fragen:
- Warum ist ECG-Input so schlecht, obwohl er bei LSTM besser war?
  - Warum ist Input "both" schlechter als wenn nur PKG präsentiert wird?
  - Bei längeren samples: Wie ~~hätte man~~ sieht Genauigkeit aus, wenn man Klassifizierung der einzelnen Fenster kombiniert?

data/results/  
sched2

PNC: Re-evaluation ~~sched~~ von Windowed-Modellen

3 modi:

- mean bildet Mittelwert aller predictions
- max klassifiziert als abnormal (1) wenn eins der Fenster als abnormal klassifiziert wurde
- hconf wählt das Fenster die Ausgabe aus, die am eindeutigsten war

Ergebnis für sched2 - ecg: ⇒ beste Ergebnisse für max,

	train	test
mean	63	58
max	74	65
hconf	76	69
base	62	62

Trotzdem ist netz schlechter als ursprüngl. LSTM

⇒ Deep-Netze werden verworfen.

22.08.2016

PNC: Evaluation der alten Netze LSTM<sub>e</sub> u. LSTM<sub>p</sub> (gemeinsam)

Idee: PkG mit LSTM<sub>p</sub> klassifizieren, EKG mit LSTM<sub>e</sub> und dann Vorhersagen kombinieren (mean, max, hconf)

results | classify

final

	only ecg	only pcg	mean	max	hconf	pcg on full dataset	min
training	70	53	63	54	63	82	69
test	74	65	74	65	74	83	74

LSTM<sub>p</sub> ist schlechter auf training-a als auf ges. Datensatz

- ⇒ Vorhersage nochmal genauer anschauen
- ⇒ 91% des trainingsdatensatzes bzw. 87% des Testdatensatzes wird als abnormal klassifiziert
- ⇒ training-a ist schlechter/schwieriger als andere Datensätze?

02.09

22.08.2016

Seltsames Verhalten von LSTM<sub>e</sub>

Beobachtung: LSTM<sub>e</sub> entscheidet sich entweder sofort nach dem ersten sample für ein Klassenlabel oder oszilliert „zufällig“.

- ⇒ Test: Klassifikation = sign(mean(ecg))
- ⇒ Accuracy Score 0.69
- ⇒ Unterschiedl. Setup/Helfer, der die Messung durchgeführt hat bei normal vs. bei abnormals?  
Vmtl. korreliert Platzierung der Elektroden zufälligerweise mit Klassenlabel.
- ⇒ Erklärt warum keins der späteren Netze (die auf normalisierten Daten trainiert wurden) so gut wurde wie LSTM<sub>e</sub>.

Nolds : Doku + Tests

7.09.2016

Kleines Update für Nolds

- Exportierte Doku mit Sphinx
- Docstrings in restructuredText umgewandelt.
- Echte unitests statt manuellen Testfunktionen

Zusammenfassung v. Computing in Cardiology 2016

21.09.2016

- atrial fibrillation ist immer noch heißes Thema
- relativ viele mathematische Modelle vertreten
  - insbes. interessant: O'hara-Rudy, Courtemanche
- programmiert <sup>z.B.</sup> in C (Backend) u. Java (Frontend)
- Medikamente werden nur über ihre Effekte modelliert, nicht biochemisch oder über Nebenwirkungen
- multiscale = single-cell models in 1d/2d/3d structure
- Wearable devices sind ein Thema, hauptsächlich photoplethysmography
- PhysioNet challenge
  - abnormal/normal Labels haben keine Korrelation zu eigtl. Herzgeräuschen in den Daten
  - Datenbank entstand durch E-mail-Betteln von Gari an alle, die in Vergangenheit Herzdaten veröffentlicht haben
  - sehr wenige Überraschungen unter Einträgen (nur sparse coding)

Nolds: Veröffentlichung von Version 0.2

14.10.2016

Features:

- HTML-Doku mit Sphinx
- unitests
- example-code ausgelagert
- code entspricht (weitgehend) PEP8-conventions

Frequenzfeatures in Test-Suite eingebaut

07.11.2016

Recherche und Lernzettel zu "frequency band power" als Jupyter-Notebook: Promotion/Notizen/PSD of frequency bands.ipynb  
 zusätzliche Features in test.py: HF, LF, LF/HF, total power

PSD of frequency bands.ipynb

11.11.2026

## Grenzwerte für Messgrößen

Recherche für Grenzwerte für Tests in der Test Seite:

- Task Force nennt nur bei der Hälfte der Werte eine Quelle (Bigger 1995).
- kaum größere Studien zu finden ( $n = 10$  bis  $n = 50$ )
- Werte aus 24 h-Recordings nicht vergleichbar mit Werten aus 5-min-Recordings.  
 ⇒ Wo möglich eher auf 5-min-supine recordings vertraut (Wir simulieren einen gesunden 30-jährigen auf einem Krankenbett.) nicht zu sportlich.
- Teilw. gute Übereinstimmung, teilw. extreme Schwankungen zw. Studien  
 ⇒ Parameter nur sinnvoll im Vergleich "vorher - nacher"?
- kaum Angaben für nichtlineare Parameter (hurst, LCE, ...)
- und bei der einen Studie stimmen die Werte dann überhaupt nicht überein (Wolf vs. Eckmann / Rosenstein)

"unrealistische" Stellen im Modell nach Parameterkorrektur:

- Bluthochdruck
- zu wenig power im Frequenzspektrum
  - wegen fehlendem Noise ?
- zu wenig chaos
- SD2 zu klein (wegen kleiner Aufnahmelänge ?)

eher nicht, SHM hat ja keine Langzeiteinfüsse

18.11.2026

Idee zur SHM-Erweiterung: EKG-Signal erzeugen

Task Force-Paper schlägt vor medizinische Geräte mit künstl. EKGs zu testen.

Idee: EKG-Synthese aus SHM (Heart austauschen, evtl. sogar mit 3D-Modell der Depolarisation)

Nolds 0.3 mit RANSAC

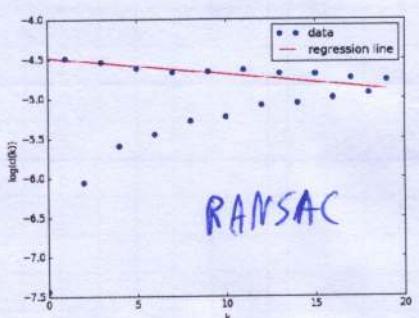
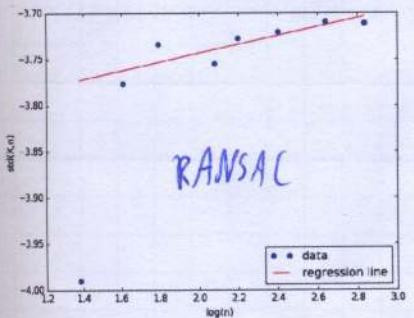
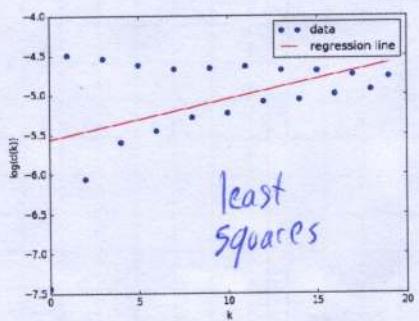
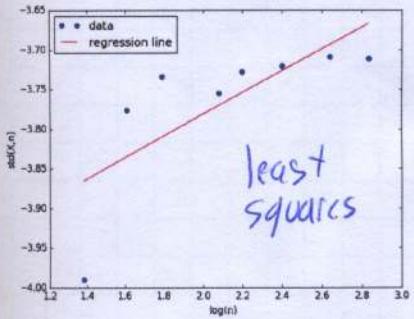
19.11.2016

RANSAC - fitting für alle Algorithmen mit Line-Fitting.

- Nominaler Hsg-Algorithmus immer noch als Option wählbar.

Einfluss auf Tests:

- Outlier bei DFA ignoriert
- einschießen auf eine Linie von zweien bei Lyap-r



Problem: Andere Plots in HRVDB  
lassen vermuten, dass  
lineare Phase sich eher  
am Anfang befindet.

⇒ Ist es eigentlich die  
flachere Linie weiter hinten  
das "abnormale"?

Dsp:

Problem: Warum haben  
die Daten überhaupt  
diese setzende Form?

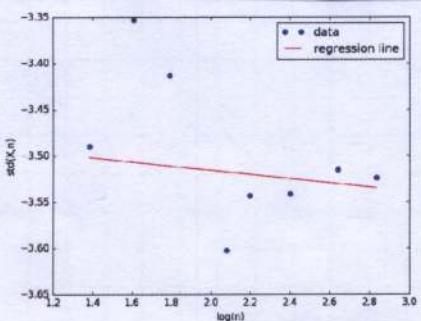
⇒ Parameter von  
Lyap-r anpassen?  
⇒ Never test auf HRVDB

DFA mit RANSAC?

19.11.2016

Beobachtung: RANSAC zum Filtern der "Frends" im DFA  
scheint keinen Sinn zu

machen, da die einzelnen Datenpunkte  
wild umher springen ⇒ lieber  
das statistisch "stabilere" Least squares  
verwenden. (Macht auch Sinn, da  
bei DFA die Fits ja auch willkürlich sind.)



23.11.2016

## Neue Parameter für lyap-r

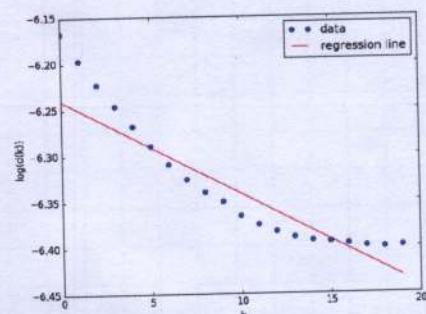
	Vorher	Nachher
emb-dim	10	10
lag	None*	1
min-tsep	None*	20

\*: chosen according some algorithm by nolds.

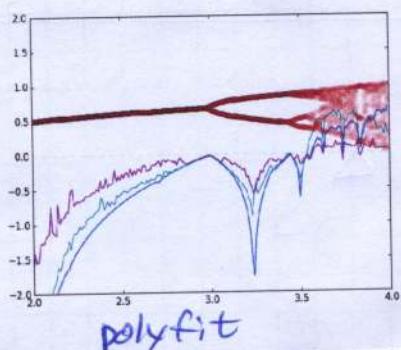
According to some study cited in Acharya 2004 (Acharya used Algorithm of Wolff)

Nach 20 Herzschlägen sollte keine temporale Übereinstimmung mehr bestehen (Mayer Waves bei grob etw. über 10 Herzschlägen)

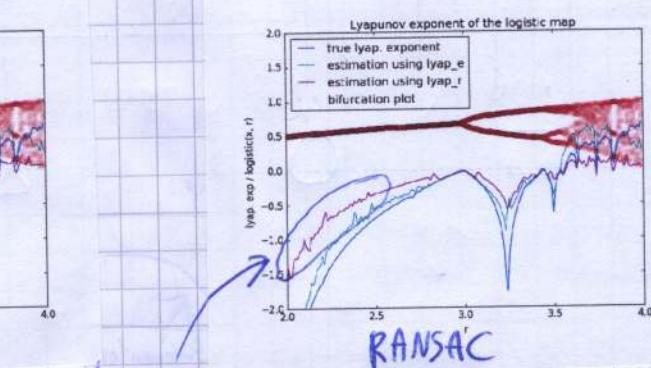
Ergebnis: Bei Test-Suite sind jetzt die seltsamen Doppel-Linien verschwunden und das Ergebnis stimmt eher mit dem von lyap-e überein



## Auswirkung von RANSAC auf plot-lyap



polyfit



RANSAC

Weichere Kurve,  
weniger Zacken  
⇒ stabiler?

## Auswirkungen von RANSAC + neuen Parameterwerten auf HRVDB-analyse

Experimente vom 26.06.2016 wiederholt mit neuen Einstellungen (auf Ladon).

Interessante Änderungen:

- corrDim, dfa, lyap-r und horst verwenden jetzt RANSAC
- lyap-r - parameter: emb-dim = 10  
lag = 1  
min-tsep = 20

Mehr ist, da  
ähnlichkeit  
scheint,  
Treppenf  
manchen  
"Treppen"

nst\_162

64600

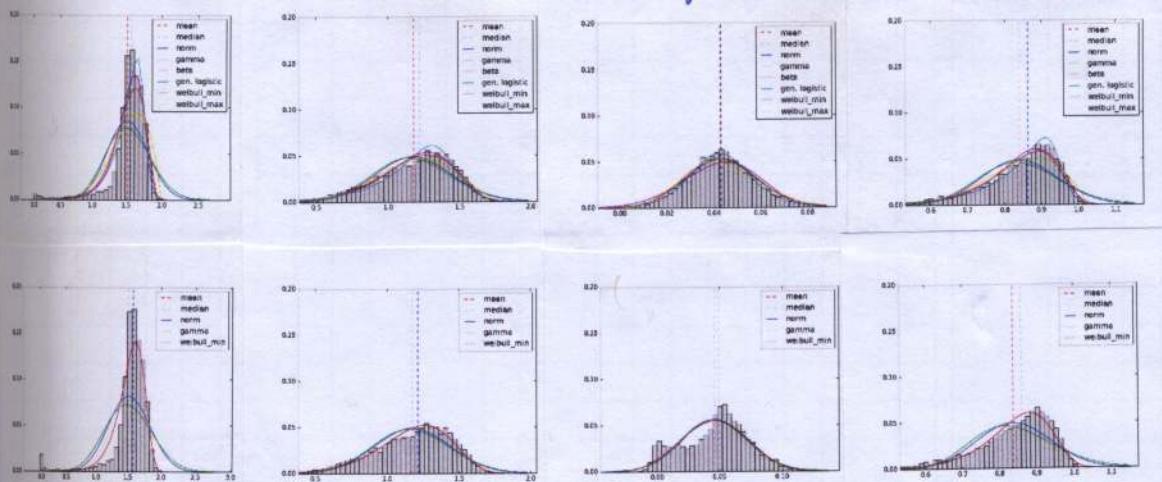
corrDim

dfa

old

lyap-r

hurst



Promotion/Results/  
chaos/nonlinear  
-measures - 7.1

new (RANSAC + adjusted parameters for lyap-r)

kaum Veränderung

kaum Veränderung

Mehr exakte Nullen. Grund dafür ist, dass es cluster von sehr ähnlichen Vektoren im Orbit zu geben scheint, so dass die "Linie" zur Treppenfunktion wird und RANSAC manchmal an so einer langen "Treppenstufe" hängenbleibt (s.u.).

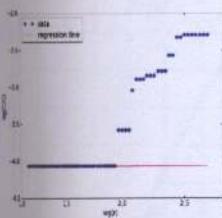
Fragen:

1. Wo kommen die Cluster her?
2. Dann lieber kein RANSAC für corrDim?

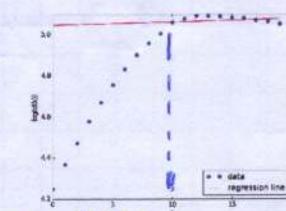
Mehr niedrige Werte. Grund dafür ist, dass die "Linie" bei lyap-r keine sind. Stattdessen handelt es sich oft um Kurven, die steil beginnen und dann abflachen.

Das scheint etwa bei einer Trajektoriellänge von 70

zu passieren (s.u.). RANSAC kann dann eine von zwei möglichen Steigungen "auswählen" per Zufall.



WT\_16273  
64600 - 64800



WT\_76273  
27200 - 27400

$\Rightarrow$  RANSAC beibehalten aber trajectory-len auf 70 setzen?

Die 70 klingt verdächtig nach der Zykluslänge der Daten (Noyer-Wellen als niedrigfrequentes Merkmal)

16.11.2016

## Beginn Paper für Modelica 2017

Erste

Zwei Paper geplant von Nico:

- MoVE, chronologisch erstes Paper
- MoIE, Nachfolgepaper, bezieht sich auf MoVE, interessanter

brau

Prei

-

Ein Paper geplant von Marcel:

- MoDE, Nachfolgepaper zu MoIE, noch nicht ganz fertig vom Inhalt her

-

Zusätzlich: Eigene Tutorial-Session für MoTE?

-

29.12.2016

## Einreichung der Paper für Modelica 2017

MoVE und MoIE eingereicht, MoDE lieber verschoben bis das Tool auch fertig ist.

Tutorial-Session wurde auch nicht angemeldet  
(zu viel Aufwand für zu wenig Gewinn)

-

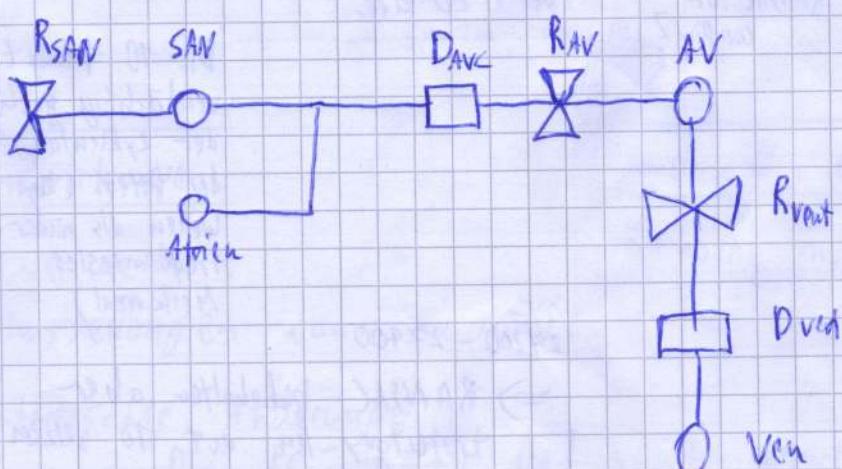
01.02.2017

Idee für Valis HRT-Modell: Modulares Kontraktionsmodell

Komponenten: Refractory Gate, Pacemaker, Delay



Lässt Signal nur durch, wenn Refraktärzeit vorbei ist.  
Generiert Signal, wenn für X Sekunden kein Signal ankam.  
Verzögert Signale.



⇒

Problem: Signal kann auch v. AV zu Atrien laufen.

-

Erste Komponentenmodelle für modulares Kontraktionsmodell fertig

06.02.2017

~~Never~~ Ordner "Schoetz Thesis" in modular-contraction brauch.

Drei Implementierungsvarianten:

- Unidirectional: Refractory Gate, Pacemaker v. Delay als unabhängige Komponenten mit einfachem Boolean - input und - output.
- Bidirectional: Refractory Pacemaker vereint Refractory Gate u. Pacemaker. Alle Komponenten haben einen Boolean - input und - output von beiden Seiten ("up" und "down" wegen physiologischer Entsprechung). Refractory Pacemaker hat Komponenten, die als "replaceable" deklariert sind.
- Bidirectional Mixin: Gleicher Aufbau wie Bidirectional, aber statt Komponenten mit Konnektoren enthält Refractory Pacemaker konnektorlose Komponenten mit "outer" Variablen, die mit "inner" Variablen im Refractory Pacemaker verknüpft sind.

Vorteil: ~~die~~ Gleichungen im Refractory Pacemaker lassen sich besser lesen.

Nachteil: Implizite Verknüpfung von "inner"- und "outer"-Variablen scheint unnötig die Verständlichkeit des Modells zu erschweren

⇒ Beste Variante wäre vlt. Bidirectional Mixin mit expliziter Verknüpfung der Variablen der Teilmodelle mit denen des Gesamtmodells ohne inner/outter.



18.08.2017

Erste Outline zum Modeling-Style-Paper

Idee: Keine der gängigen Sprachen ist geeignet, um echte Multi-scale-Modelle von höherer Komplexität zu bauen.

Paper/eigenes/  
2017-Modelling-Style

Gewünschte / benötigte Eigenschaften:

- deklarativ (SBML, CellML)
- menschenlesbar (MML)
- hierarchisch (2, Modelica)
- modular (2, Modelica)

Erklärung der Probleme u. Anforderung am Beispiel von ST - Kontraktionsmodell.

Zielpublikum: Ärzte, die Modelle schreiben; Systembiologen

6.09.2017

Hautverkehr mit Physiome-Wissenschaftlern

J. Bassingthwaite:

- Kompatibilität zu SBML / CellML / JSim
- Kompositionstool SemCom
- Kommt Modelica mit PDEs klar?
- JSim is 300 times as fast as Matlab on most tasks  
⇒ Was ist mit Modelica?
- keine Richtige Antwort darauf, wie Physiome das Problem mit großen Multiscale-Modellen löst

Frage: Wie wird Komplexität gehandhabt?  
Warum JSim / MML statt Modelica?

D. Noble → P. Hunter:

- Physiome-Tools: OpenCOR, OpenCMIS
- bottom-up approach will never succeed
- units are important, bond graphs can help
- modularity and declarative languages are the only way
- FieldML als Weiterentwicklung von CellML
  - soll komplexere zusammengesetzte Modelle ermöglichen
  - weiß nicht viel über Modelica, findet es aber wichtig
- Argument gegen Modelica in Physiome: Do not use projects that have commercial aspect  
⇒ Physiome-Software soll frei für wissenschaft und kommerziellen Gebrauch sein.

Suche nach passenden Journals für Modelling-Style-Paper 07.09.2017

Systematische Suche über Thompson Reuters nach ~~Modellierung~~ Kardiologie und Systembiologie + Ranking der Journals nach thematischer Nähe (laut ~~"aims & scope"~~ auf der Webseite).

Promotion / Notizbuch  
journals.txt  
2017-Modelling-style/  
Journals.xls

Favorites (Beispielhaft):

- Circulation Research (sehr hoher IF, passt recht gut)
- Wiley Interdisc. Reviews: Systems Biology and Medicine (kleiner IF, thematisch ideal, aber ~~evtl.~~ evtl. nur Reviews?)
- Cell Systems (hat noch keinen IF, wird aber vermutlich sehr hoch, thematisch wie Circ. Research)

Modelica vs Jsim: Hodgkin-Huxley

13.09.2017

Übersetzung der Jsim-~~VBA~~ Implementation von HH nach Modelica, um Geschwindigkeit zu vergleichen:

Solver: RK4

Anzahl Schritte: 3.000.000

Zeit Jsim: 43s

Zeit Modelica: 53s (+ 20s schreiben der Output-Datei)

Modelica vs Octave: Hodgkin-Huxley

15.09.2017

Zeitmessung mit Octave für HH-Implementation von Dennis (hh1952):

Solver: RK4

Anzahl Schritte: 300.000

Zeit: 343,51s  $\Rightarrow$  Faktor 100 langsamer als Modelica/Jsim

ABER: Octave hat keinen JIT  $\Rightarrow$  Faktor 100 zw. Octave und Matlab ist bei großen Schleifen normal.  $\Rightarrow$  Muss nochmal mit Matlab wiederholt werden

25.10.2017

## Hodgkin-Huxley: Zeitmessung mit Matlab

Note

Zeitmessung v. Dennis bh 1952 - Modell mit Matlab:

Auf

Solver: RK4

Fun

Anzahl Schritte: 3.000.000

grä

Zeit: 7,4 s  $\rightarrow$  Faktor 5 schneller (!) als Modelica/J

-

Version: R2017b

-

~~Probsteier~~

-

Wiederholung der HH-Zeitmessung mit  $\Delta t = 0.01$ 

-

Problem: Erste Zeitmessung wurde mit  $\Delta t = 0.1$  durchgeführt. Modelica liefert hier bei RK4 zwar einen korrekten Plot, aber alle anderen Tools liefern fehlerhafte Daten (oder stürzen sogar ganz ab).

j

$\Rightarrow$  Wiederholung des Experiments mit  $\Delta t = 0.01$  und simulationTime = 3000 ms

Das

von

beschr

bietet

Octave	3164 s	(hochgerechnet)
OpenModelica	54 s	
JSim	28 s	
Matlab	13 s	

$\Rightarrow$  Modelica ist langsamer als JSim, aber Matlab ist schneller als beide.

~~Ziel Erreichen~~Zusätzlich Test von OpenCOR: 2,5 s!  $\Rightarrow$  Schnellstes Ergebnis.Zur Fairness weiterer Test mit besseren Solvoren für JSim und Modelica mit  $\Delta t = 0.1$  und simulationTime = 3000,

Vergle

mit

~~alles~~

JSim	Radau	15 s
OpenModelica	dassl	6.7 s

$\Rightarrow$  Unter dem Strich kann Open Modelica gut mithalten, da bei besseren Solvoren die Schrittweite vergrößert werden kann. (Schlägt OpenCOR aber nicht.)

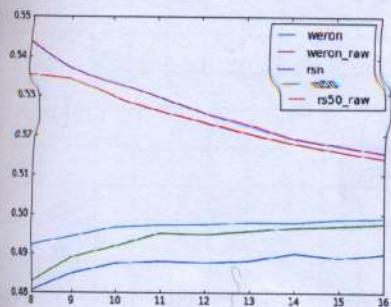
Außer

mit d

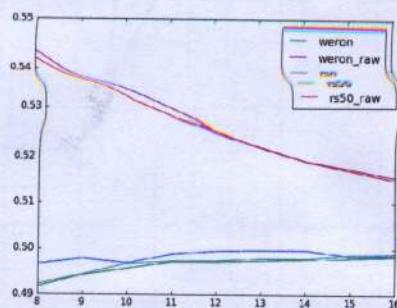
Aufgrund eines Github-Issues habe ich mir die Funktion `hurst_rs` angesehen und ein paar Dinge geändert:

- Der Amis-Lloyd correction factor ~~ist~~ wird nun per default auf das Ergebnis angewandt
- Parameter `nvals` nimmt in der Default-Einstellung nur n die größer 50 sind.
- Die Standardabweichung für die R/S-Werte basiert jetzt auf der erwartungsfreien ~~standard~~ Stichprobenvarianz ( $\frac{1}{n-1}$  statt  $\frac{1}{n}$ ).

Das Resultat: `hurst_rs` kann jetzt sehr gut die Plots von Weron ~~(2002)~~ nachbilden, sowie die exakt beschriebenen Werte für brown72 (wobei dafür die biased-Version der Stdabw. verwendet werden muss).



Vergleich nolds(rs50, rs50\_raw)  
mit Weron (weron, weron\_raw)  
~~mit~~ mit biased-version



Vergleich nolds(rs50, rs50\_raw)  
mit Weron (weron, weron\_raw)  
mit unbiased-version.

Promotion/results/  
chaos/hurst-weron  
↳ length-test-newn-  
with-reported-exp  
↳ length-test-newn-  
with-reported-ssfd-  
nofull.png

Außerdem hat nolds jetzt ein "datasets"-Modul mit den folgenden Datensätzen:

- brown72: Testdatensatz von [bearcave.com](http://bearcave.com)
- tent-map: Generator für die Tent map (chaotische Zatr.)
- logistic-map: Generator für Logistic map (vorher in plot-type)
- Fbm: Fractional Brownian Motion (wurker in measures)
- fgn: fractional Gaussian noise
- drandom: Echter Zufall mit quantumrandom

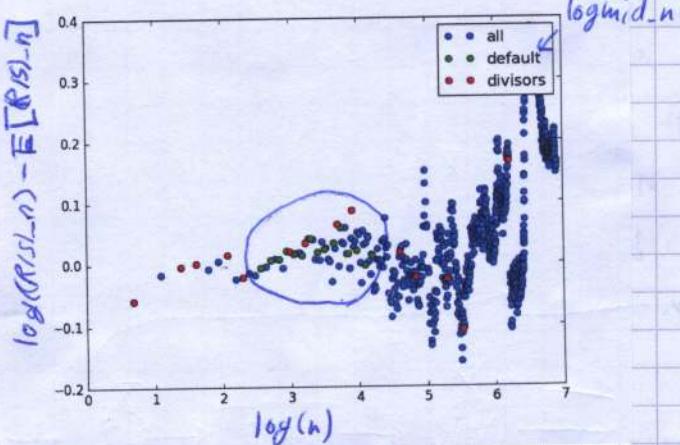
5.1

30.11.2017

## Weitere Verbesserung der Default-Parameter für horst\_rs

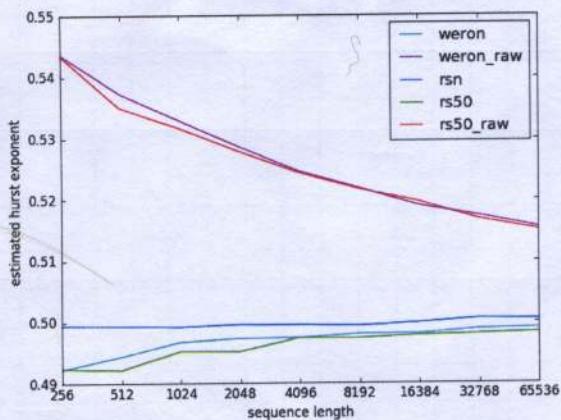
Idee / Beobachtung: Plotzt man den Graphen für alle möglichen  $n$  bei horst\_rs zeigen sich oft Avirektor bei sehr niedrigen  $n$  (in Form eines eher Parabelförmigen Verlaufs) und bei sehr hohen  $n$  (in Form von Schwingungen)  $n$ . Die "richtige" Steigung sieht man am besten in der Mitte des Plots.  
 ⇒ Neue Funktion logmid\_n, die gleichmäßig verteilte  $n$  aus dem logarithmischen Plot aus der Mitte auswählt.

Promotion/results/chaos/  
hurst\_weron/  
compare\_nvrs.py



Ergebnis für Weron Figure 2: Deutlich bessere Werte

Promotion/results/chaos/  
hurst\_weron/  
length-test-logmid.py



14.12.2017

Idee: Ontologie in Interface-Library übersetzen

Paper von Physiome-Wissenschaftlern argumentiert, dass Interfaces ad-hoc auf Basis semantischer Informationen gebildet werden müssen.

Eine Umsetzung in Modelica wäre möglich, wenn man die Ontologiebegriffe als Interfaces / Datentypen definiert.

## BooleanDelay - Modell für Contraction im Modelica

14.12.2017

Nach nochmaligem Durchsehen der Contraction "Library" in Schöelzel Thesis und einer Recherche in den Modelica-Foren habe ich doch eine Möglichkeit für ein echtes Delay für diskrete Variablen gefunden.

Die Lösung bedient sich einer zusätzlichen kontinuierlichen Variablen. Beispiel:

```
Boolean input signal;
Boolean delay;
Real cont;
```

preU ist nötig um Events  
zu erkennen

equation

$$\text{cont} = \text{if preU(signal) then } 1.0 \text{ else } 0.0;$$

$$\text{delayed} = \cancel{\text{delay}}(\text{cont}, 0.3) > 0.5;$$

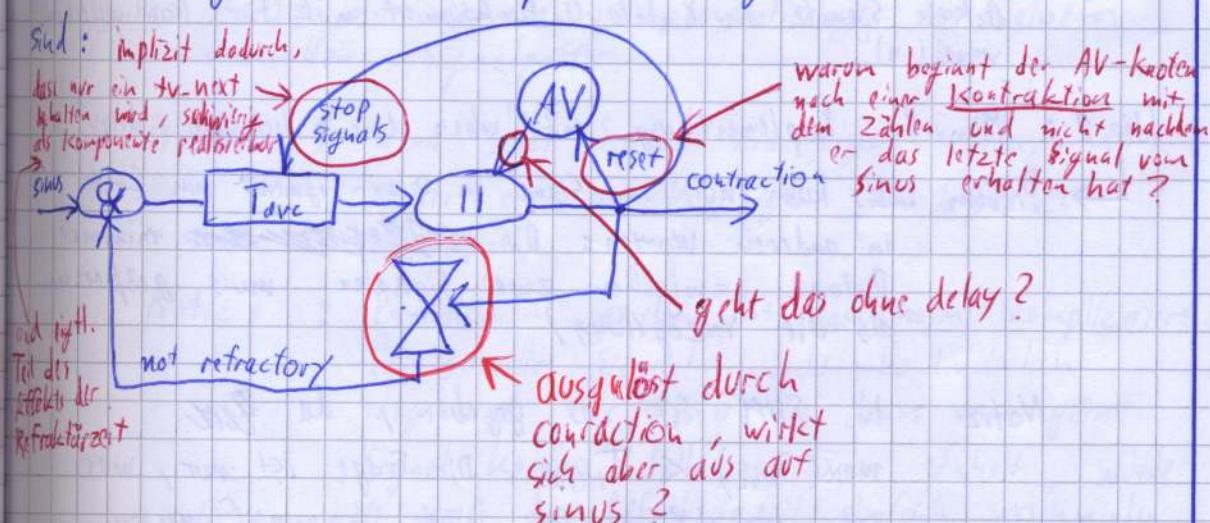
## Korrektur von Contraction2 im SHM

Durch das Einfügen einiger Aufrufe von preU funktioniert die einfache straightforward-Implementation Contraction2 jetzt auch in OpenModelica und liefert exakt die gleichen Ergebnisse wie der Workaround in Contraction.

## Schematische Darstellung des Kontraktionsmodells im SHM

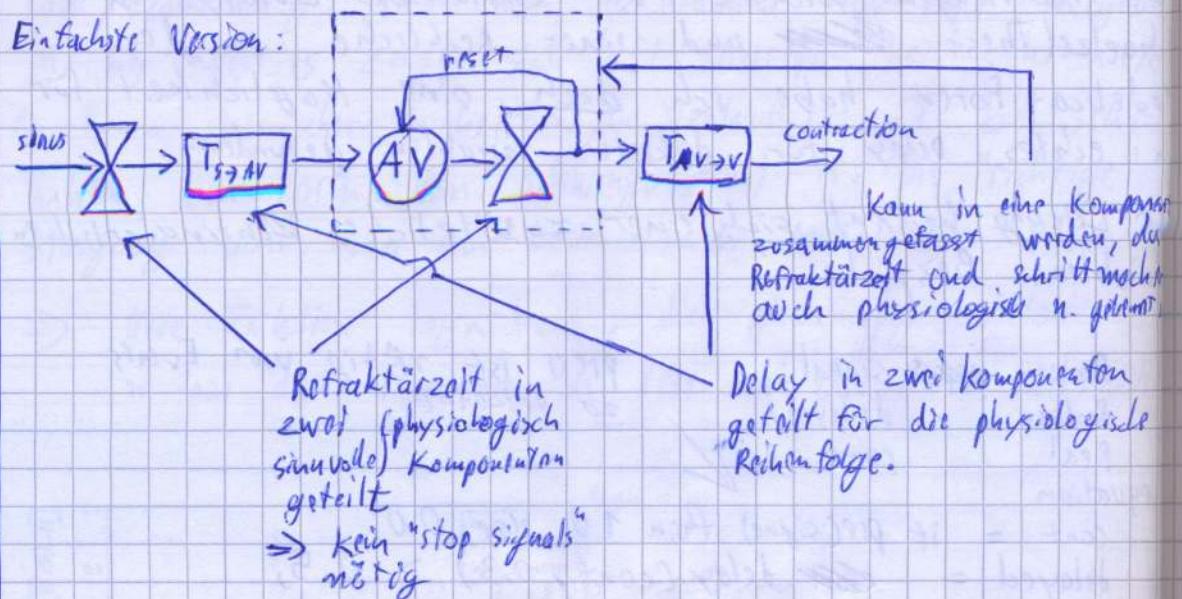
19.12.2017

Versucht man die Kontraktion im SHM zu modularisieren, fällt auf wie ungeordnet die Komponenten eigentlich verbunden sind:

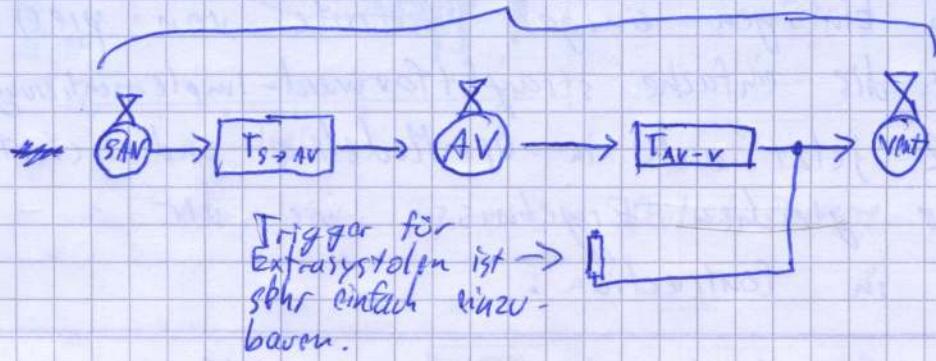


Mit den gleichen Komponenten ( $\textcircled{O}$ ,  $\textcircled{X}$  und  $\square$ ) lässt sich ab Idee ein weitaus physiologisch sinnvolleres Kontraktionsmodell erstellen.

Einfache Version:



Korrektste Version (mit  $\textcircled{X}$  als Schrittmacher mit Refraktärkomponente)  
Entspricht dem physiologischen Aufbau



18.01.2018

Verschiedene Delay-Varianten für Unidirektionale Komponenten

Problem: Modelica bietet keine Standard-Komponenten / -funktionen für diskrete Signale an (`delay()` funktioniert nur für kontinuierliche Variablen).

Idee 1: Manuelle Implementierung mit "when time > last.signal + delay"

$\hookrightarrow$  Problem: Es kann nur ein Signal im Delay-Segment sein (oder in anderen Worten: Die ~~minimale~~ minimale Distanz zwischen zwei Signalen muss größer sein als die Verzögerung)

Notiz: In SHM ist das gegeben, da ~~max~~  $\max(T_{\text{ave}}) < \text{Trefrac}$ . Die Frage ist nur, was bei anderen Parametern bzw. Kombinationen passiert.

Idee 2: Wir nutzen `delay()` indem wir das diskrete Eingangs-signal in ein kontinuierliches umwandeln:



Wichtig: Wir brauchen die Stufenfunktion, damit das Signal die Diskretisierung in history-arrays überlebt, die bei `delay()` passiert.

SHM/Schreitfibrillation  
Components /  
Contraction /  
Unidirectional /  
Built-in Delay

→ Probleme: - Wenn das Delay vergrößert wird, sieht man einen Teil des Signals doppelt  $\Rightarrow$  Es kommen mehr Signale an als abgesendet wurden.

- Auch wenn das Delay verkleinert wird, kann durch die Veränderung der Ausgangswert "spontan" von 0 auf 1 oder anders herum springen  $\Rightarrow$  Ein Ausgangssignal entsteht an einer Stelle, die der gar keine Flanke im kontinuierlichen Signal war.

Außerdem können so auch Events "überspringen" werden.

- Verändert sich das Delay, wirkt sich dies sofort auf alle in der Komponente befindlichen Signale aus. Das ist physiologisch nicht sinnvoll (eine Ventrikelzelle weiß ja nicht, was die "erste" AV-Zelle gerade tut.)

Idee 3: Zurück zur manuellen Implementierung mit einem manuellen Signalpuffer.

→ Problem: Wenn jeder Signal sein eigenes Delay hat, könnte ein schnelles Signal ein langsames überholen.

SHM/Schreitfibrillation/  
Components /  
Contraction /  
Unidirectional /  
Manual Delay

Notiz\*: Physiologisch gesehen ist das noch komplizierter. Ein schnelles Signal würde kontinuierlich langsamer, je näher es dem vorherigen Signal kommt. Falls es dieses einholt, würde es wegen der Refraktärzeit einfach verschwinden.  $\Rightarrow$  Dieses (letzte) Verhalten lässt sich leicht implementieren, das "Langsamwerden" ignoriere ich.

Weiterer Vorteil: Das Verhalten ist leicht für das bidirektionale Modell verwendbar.

Implementierung: Da es kein "Überholen" gibt, können die Ausgangszeiten einfach der Reihe nach in einem Array eingebracht werden.

Das erste Element ist immer das nächste Event  $\Rightarrow$  Wenn die ~~die~~ Wertezzeit für dieses abgelaufen ist, muss man nur alle anderen Einträge nach links verschieben.

### Tatsächliche Einflüsse auf AV-Delay

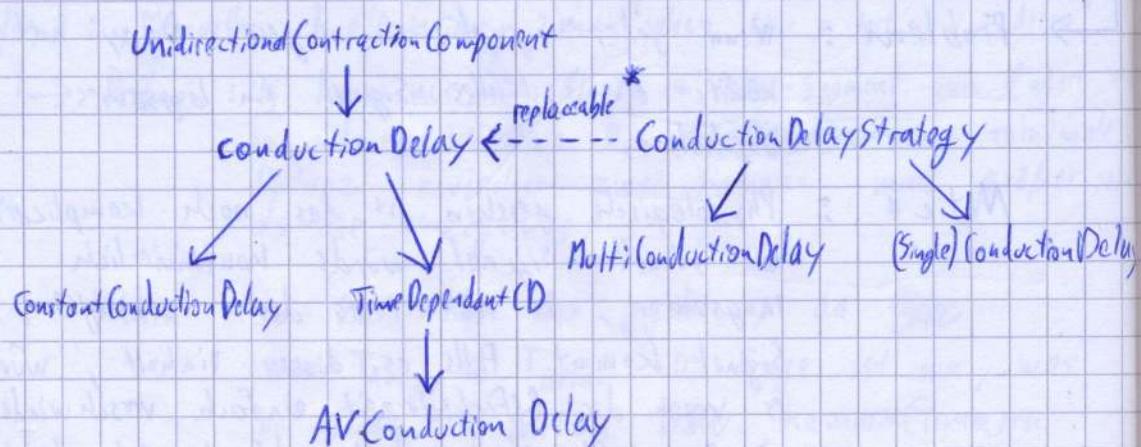
Recherche: Wie ist das eigentlich physiologisch mit dem AV-Delay? ~~Die~~

Interessanterweise geht es dabei eigentlich nur um den Delay zwischen AV-knoten und His-Bündel. Auswirkungen haben (v.a.) die folgenden Effekte:

- Autonomes Nervensystem } Existiert im STHM, wirkt sich constant/constant auf Tarc aus
- Zirkulierende Catecholamine } abt nicht auf Tarc aus
- Recovery - Effekt  $\leftarrow$  als einziger im STHM
- Facilitation - Effekt } existieren noch nicht
- Fatigue - Effekt

Interessantes Modell: Billiette + Nattel 1994  
 $\hookrightarrow$  Talajic et al. 1991

23.01.2018 „Klassendiagramm“ für Idee 3 für das AV-Delay



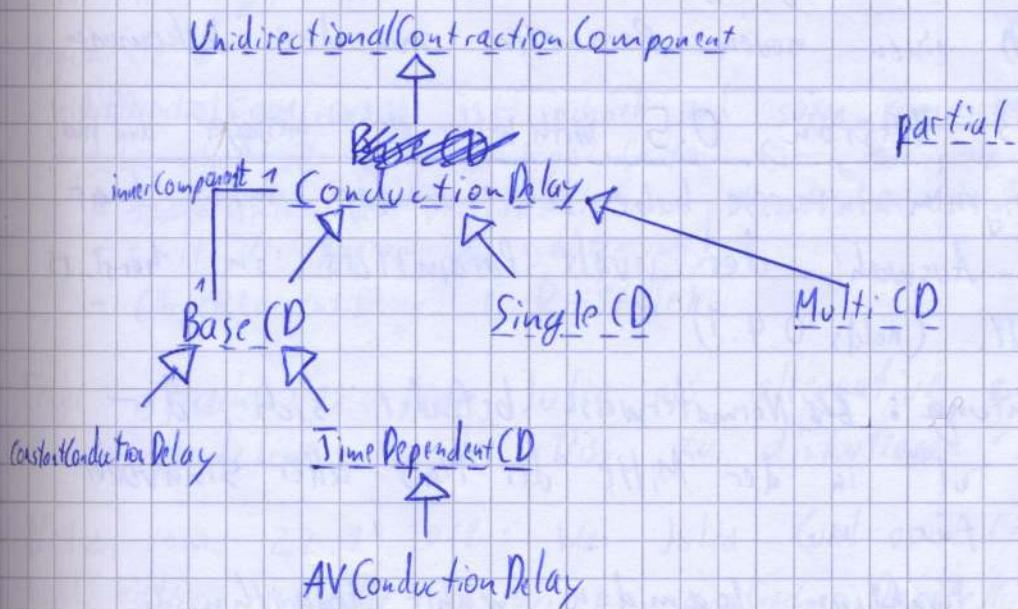
Conduction Delay delegiert an eine innere austauschbare Komponente vom Typ Conduction Delay Strategy.

\*: Es gibt zwei mögliche Implementierungen, die sich im Wesentlichen darin unterscheiden, was der Endbenutzer im Code angeben muss, um die Strategie zu wechseln:

### ~~Modellabstraktion~~

- AVConductionDelay (redeclare MultiConductionDelay strategy)
- AVConductionDelay (redeclare model Strategy = MultiConductionDelay)

Nur Modellnamen und zusätzliche Basisklasse für AV-Delay 25.01.2018



zur vereinfachten Darstellung kann TimeDependent(D) auch von Single(D) erben. Im Code ändert sich nichts außer dem extends.

### Korrektur des AV-Delay ( $A_1, A_2$ vs $H, A_2$ )

In der Literatur (z.B. Billette et al. 1994) wird die "recovery time", die für die Länge des AV-Delays maßgeblich ist, nicht durch die Zeit zwischen zwei "eingehenden" Signalen vom Atrium ( $A_1, A_2$ ) sondern durch die Zeit zwischen dem letzten ausgehenden und dem aktuell eingehenden Signal ( $H, A_2$ ) abgeschätzt.

⇒ TimeDependent(D) muss angepasst werden (d.h. auch, dass seitdem Recht hatte mit der klinischen Zeitmessung)

Für Refractory Gate wird weiter  $A_1, A_2$  verwendet (weil physiol. sinnvoller). Da die Wirk von 0.22 (nicht 0.9 wie im Beitrag vom 18.01. aufgetragen!) ⇒ Es könnten doch zwei Signale im Delay stecken aber

auf dem  $H_1 A_2$ -Intervall aufbautes muss  $T_{\text{refrac}}$  von interessierter Komplexität die "mittlere" conduction time erhöht werden.  
In Erwartung besserer Werte habe ich daher  
 $T_{\text{refrac}} = 0.335$  gesetzt.

Zusätzliche kleine Korrektur: statt das nur "Überholende" Signale zu MultiCD ignoriert werden, werden jetzt auch diejenigen ignoriert, die beim Austritt weniger als  $T_{\text{refrac}}$  vom Vorgänger entfernt wären. Dazu hat MultiCD einen neuen Parameter min-dist bekommen.

24.10.2018

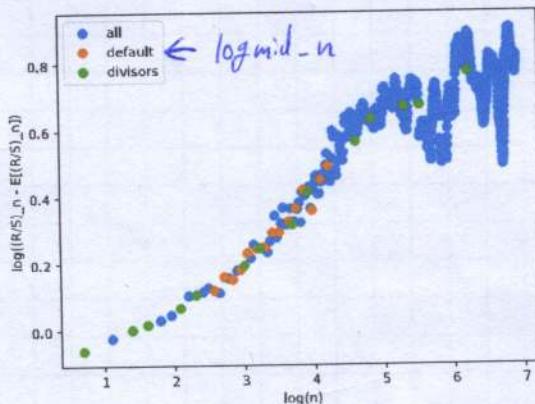
Nolds Version 0.5 with better error messages and nvals

Aus eigenem Interesse habe ich weiter mit der default-Auswahl des nvals-Parameters in kurzs gespielt. (nolds 0.4.1)

Beobachtung: ~~Bei~~ Normalerweise befindet sich der "lineare Teil" in der Mitte des Plots aller sinnvollen Werte für  $n$ .

Idee: Funktion logmid-n wählt logarithmisch verteilte Punkte aus, die genau in dieser Mitte liegen.

Ergebnis (Beispiel):



Daten: nolds.fgn(1000, H=0.75)

$H_{\text{all}} = 0.621$

$H_{\text{default}} = 0.714$

$H_{\text{divisors}} = 0.668$

Insges: Wahl scheint gut zu funktionieren.

Issues auf GitHub haben mich auf zwei Probleme aufmerksam gemacht:

- nolds funktioniert nicht mit pandas-dataframes  
⇒ Nutze np.asarray für inputdaten wo möglich
- lyap-r (und lyap-e) sind nicht sehr transparent,  
was die minimale Anzahl von datapunkten angeht

Interessanterweise sind die Formeln dafür relativ kompliziert  $\Rightarrow$  Ich habe Funktionen geschrieben, die diese berechnen

$$\text{lyap\_r\_len} : (\text{emb\_dim}-1) \cdot \text{lag} + \text{trajectory\_len} + \text{min\_tsep} \cdot 2 + 1$$

$$\text{lyap\_p\_len} : \text{emb\_dim} + \frac{\text{emb\_dim}-1}{\text{matrix\_dim}-1} + \text{min\_tsep} \cdot 2 + \text{min\_nb}$$

Tipp von Franz Cemal: Julia statt Modelica?

1.11.2018

Erfahrung von Franz: Matlab ist wegen Lizenz zu schwierig  
 $\Rightarrow$  Wechsel nach Modelica  $\Rightarrow$  Modelica bietet zu wenige Möglichkeiten für Finetuning / erlaubt manche Modelle nicht  $\Rightarrow$  Wechsel nach Julia

Argumente:

- DifferentialEquations.jl ist schnell und super komfortabel
- Julia hat alle wichtigen Eigenschaften für Multiscale-Modelle
- ~~kein~~ Declaratives Verständnis von mathematischen Gleichungen (mit Metaprogramming, schätze ich)
- Objektorientierung (R-style)

Frage: Macht es Sinn Julia als Alternative zu Modelica in der Diss zu diskutieren?

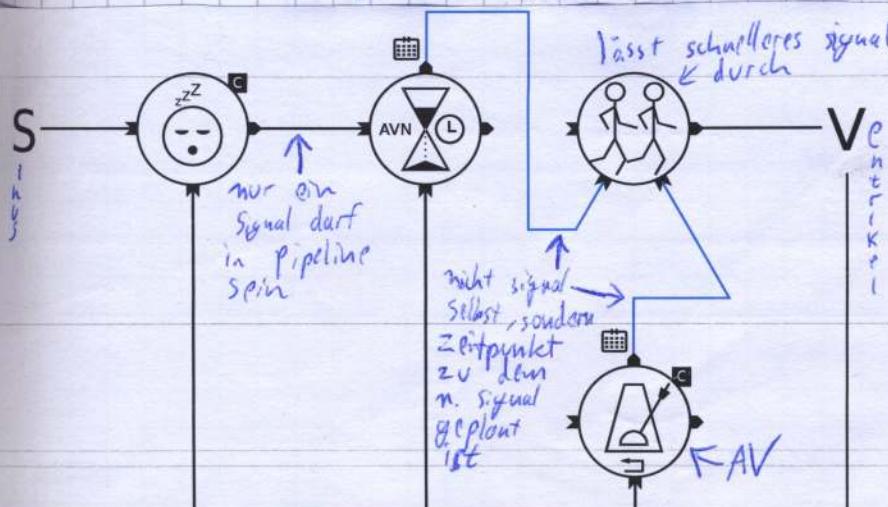
Notiz vom 22.11.2018: Was Julia (und auch CellML<sup>2</sup>) nicht bieten ist die integrierte grafische Darstellung der Komponenten und ihrer Verbindungen.

Vorbau des originalen Signalweiterleitung des STH mit Komponenten

22.11.2018

Nachdem ich fast ein volles Semester nicht in meiner Code geschaut habe, habe ich ~~schon~~ mich gefragt, wie eine exakte Darstellung der Signalweiterleitung im STH in komponentenorientierter Form aussehen müsste.

Dabei kam das folgende Diagramm heraus:



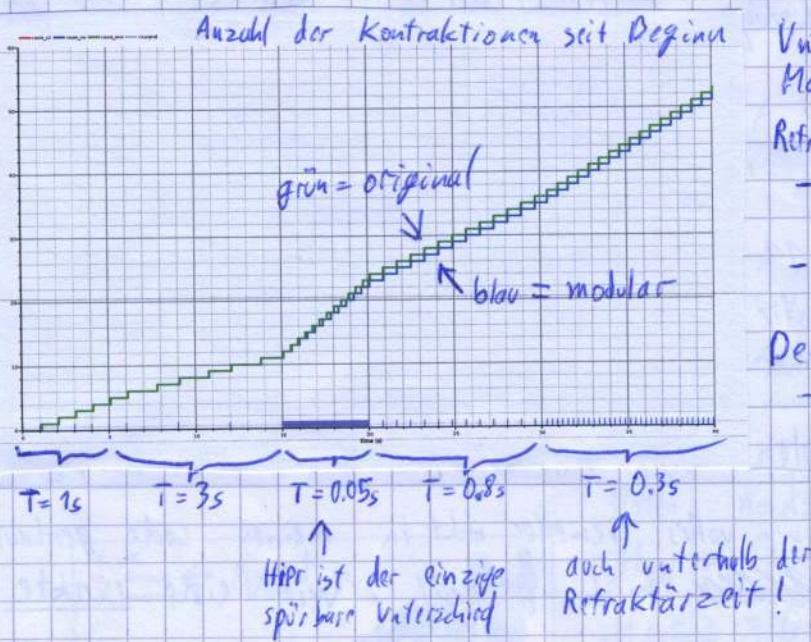
ScholarThesis/  
Components/Contraction/  
Unidirectional/  
OriginalSTM

~~Ergebnisse~~ Danach habe ich die entsprechenden Komponenten in Modelica implementiert.

Erkenntnisse:

- das Original ist zu verknüpft, um sauber in Komponenten abbildungbar zu sein  
⇒ Vereinfachung macht Sinn
- der originale C-Code trickst bei der Refraktärzeit
  - eigentlich würden sehr schnelle Signale mehrfach durch den "Refraktär-check" gehen, bis das erste wieder durch das Delay hindurch ist und eine Kontraktion auslöst
  - aber Seidel registriert nur ein Signal zwischen zwei Kontraktionen  
⇒ Vorhofflimmern wäre unmöglich zu simulieren (z.B.)
- das Nachimplementieren des Originals ist extrem fehleranfällig weil unintuitiv

Vergleich von originalem und modularem Kontraktionsmodell



Unterschiede der Modelle:

Refraktärzeit:

- modular: Zeit zw. eingehenden Signalen
- orig: Zeit zw. Kartei und eingehendem Signal

Delay:

- modulares Modell kann theoretisch mehrere Signale auf Delay halten

Ergebnis: Sauber modulär getrenntes Modell verhält sich sehr ähnlich zu Original. Unterschiede gibt es nur bei extrem schnellen Eingangssignalen.

Grund: Refraktärzeit muss bei beiden Modellen ordnen Wert haben ⇒ Übereinstimmung exakt zu machen ist schwierig, vielleicht sogar unmöglich.

31.01.2019

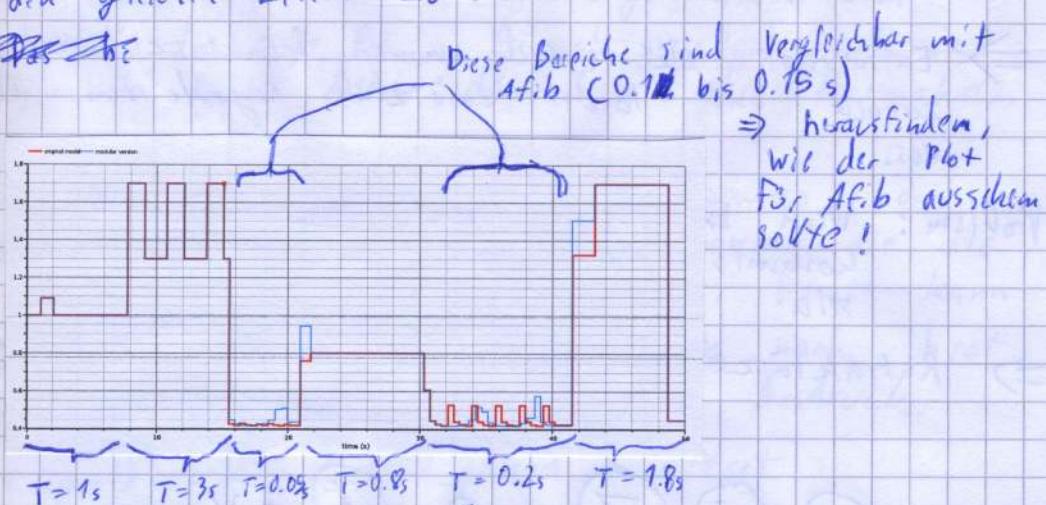
Vergleich beider Kontraktionsmodelle über RR-intervalle

Erkenntnisse beim Schreiben des Papers:

- RR-Intervalle eignen sich besser zum Vergleich der Modelle
- im Originalmodell wird für die Refraktärzeit geprüft ob  $(t - t_{\text{rest}}) > T_{\text{refrac}}$  ⇒ Mit  $t_{\text{rest}} = t_{\text{sin}} + T_{\text{delay}}$  wird das zu

$t - (t_{sin} + T_{delay}) > T_{refrac}$  und das kann man umstellen zu  $t - t_{sin} > T_{refrac} + T_{delay}$ . Wenn man die  $t_{sin}$  statt  $t_{vent}$  als Referenz nimmt, muss man die (mittlere) Delay-Dauer ~~abziehen~~ zur Refraktärzeit addieren um den gleichen Effekt zu erhalten.

~~Pass the~~

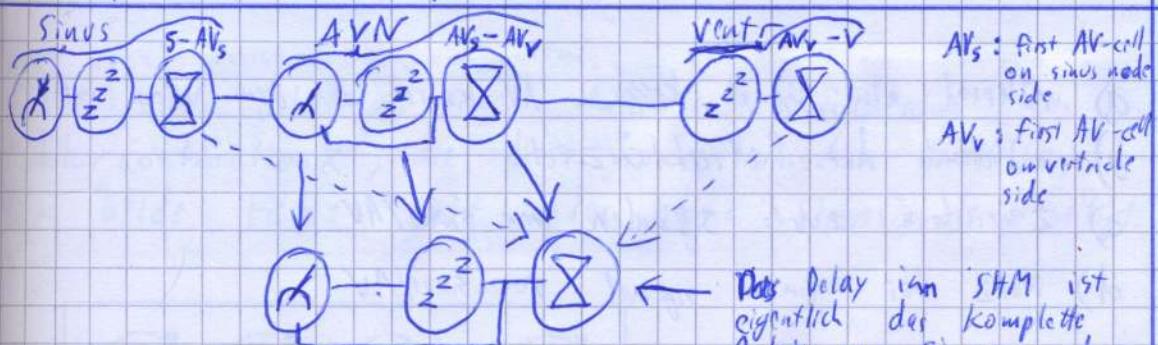


Unterhalb der Refraktärzeit unterscheiden sich die Modelle. Beide schwingen, aber das modulare Modell hat eine geringere Frequenz ~~und~~ und eine höhere Amplitude.

Erklärung:  $T_{delay}$  ist zeitabhängig, d.h. die Prüfung auf Refraktärzeit ist es im originalen Modell auch. Im modularen Modell nehmen wir den Durchschnitt und halten die Refraktärzeit damit konstant.

Frage: Was ist physiologisch sinnvoller? Eine variable oder eine feste Refraktärzeit?

### Exakte physiologische Entsprechung der Kontraktionskomponenten



Physiologisch:



SAN  $\rightarrow$  AVNs  $\rightarrow$  AVN<sub>v</sub>  $\rightarrow$  Ventricles

$$\begin{aligned} \text{4 cm} @ 0.5 \frac{\text{m}}{\text{s}} &\Rightarrow 0.08 \text{ s} \\ \text{5 mm} @ 0.05 \frac{\text{m}}{\text{s}} &\Rightarrow 0.1 \text{ s} \\ \text{5 cm} @ 2 \frac{\text{m}}{\text{s}} &\Rightarrow 0.025 \text{ s} \end{aligned}$$

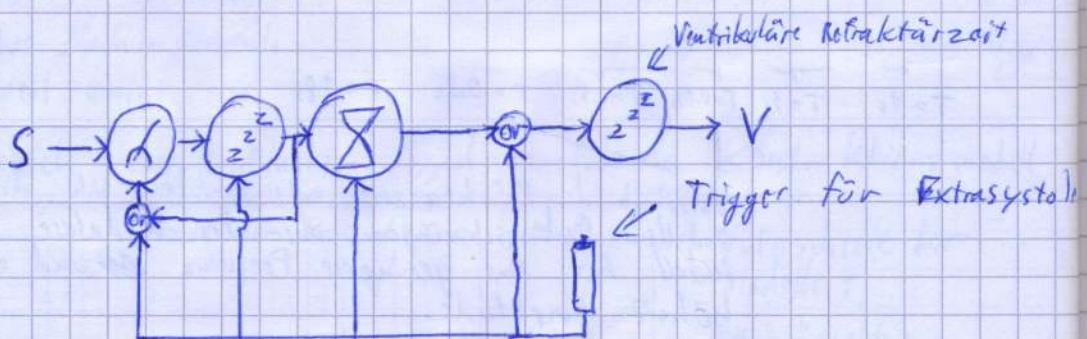
## Extrasystole im einfachen unidirektionalen Kontraktionsmodell

Idee: Ectopic beat kann entweder ein entgegenkommendes Signal auslösen oder er setzt die Timer für Refraktärzeit und Schrittmacherfunktion des AV knotens zurück. Da ~~die Refraktärzeit~~ kurz ist, ist letzteres wahrscheinlicher.

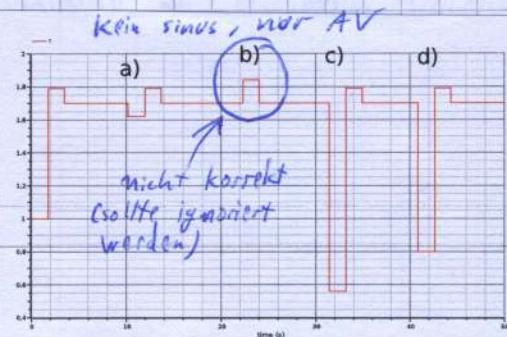
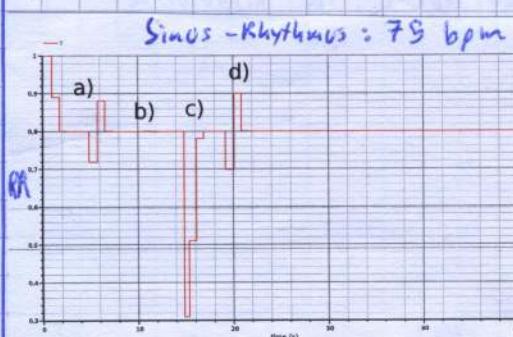
⇒ Extrasystole setzt einfach immer Refraktärzeit und Schrittmacherfunktion zurück und löscht zusätzlich Signale im Delay aus.

Problem: Wenn Extrasystole direkt nach normalem beat kommt, können beide bei nah beieinander sein.

⇒ Refraktärzeit der Ventrikel zusätzl. beachten.



Resultate:



- während ein Signal ~~kommt~~ im AV-Knoten auf dem Weg ist
- während der Refraktärzeit
- zwischen zwei Signalen von Sinus/AV
- kurz vor einem Signal von Sinus/AV

Bessere Ergebnisse sind nur mit bidirektionalen Kompensationen realistisch.

01.03.2019 Fehler im SHM: Zeitabhängigkeit von Trefrac

Erkenntnis aus der Literatur (Mendez 1956): Refraktärzeit sinkt bei Erhöhung der Herzfrequenz bzw. Senkung der Zyklusdauer.

cycle length ↓ → refractory ↓ → : Realität

delay ↑ ---> refractory ↑ ---> : SHM

Effekt im SHM  
→ geht in die falsche Richtung

# Verbesserung des modularen PVC-Modells

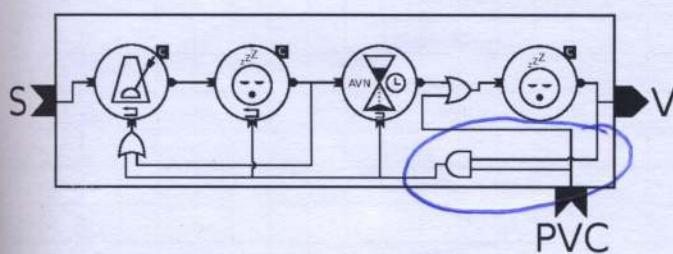
08.05.2019

Problem 1: Simulation von ExtraSystole Example war ungenau für `with_sinus = false`. Um PVC nach dem Xten beat auszulösen habe ich  $1.7 \cdot X + \text{offset}$  gerechnet. Das stimmt nur für die erste PVC. Danach verschiebt sich ja der Beginn des Zyklus durch den Reset des Pacemakers.

Lösung: Nutze die Variable `count`, die sowieso schon existiert und definiere eine Variable `t_passed`, die die Zeit seit der letzten Kontraktion zählt. Damit kann PVC präzise offset Sekunden ~~stetig~~ oder nach beat mit `X` gesetzt werden mit dem Ausdruck `pre(count) == X` und `t_passed > offset`

Problem 2: Nachdem das Timing korrigiert wurde, wurde klar, dass auch für `with_sinus = true` ein ungewollter Effekt entsteht, wenn eine PVC während der Refraktärzeit der Ventrikel ausgelöst wird. Das geschieht durch den Reset der Delay-Komponente, der zu einer (minimalen) Veränderung der Delay-dauer für das nächste Signal führt.

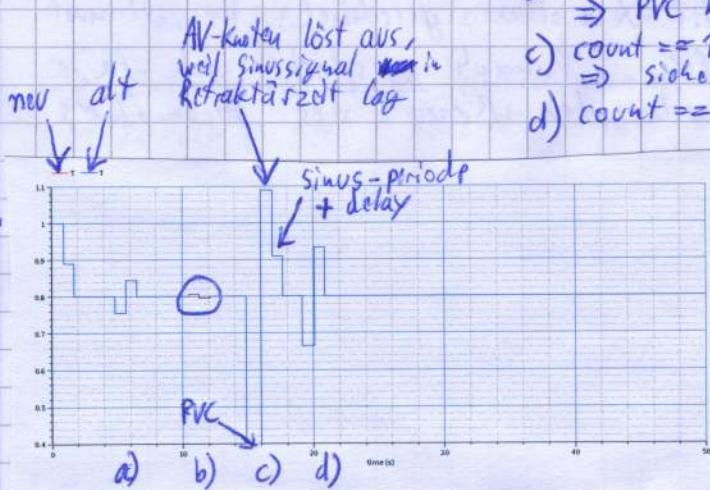
Lösung: Mir ist aufgefallen, dass man mit einer einfachen Und-Vereinigung abfragen kann, ob die Ventrikel refraktär sind. (Sie sind nicht refraktär, wenn eine PVC auch gleichzeitig zu einem Output des Refractory Gate führt.) Das führt zu einer einfachen Veränderung von `ModularContractionX`, die dieses unrealistische Verhalten für beide Fälle mit oder ohne Sinussignal beseitigt.



Code / SHM / src /  
img / Print Modular  
ContractionX Paper.pdf

## Erklärung der neuen Plots von Extra Systole Example Ausl

results/modular-contraction/  
compare-extra-before-  
-after.svg



- a) count == 5, t-passed > 0.8 - 0.05  
⇒ PVC ersetzt Sinussignal im Delay
- b) count == 72, t-passed > 0.15  
⇒ PVC kann n. ausgel. werden, da vent. refraktär
- c) count == 78, t-passed > 0.4  
⇒ siehe Bild
- d) count == 23, t-passed > 0.8 - 0.15  
⇒ PVC wird vor Sinussig ausgelöst, das dann in die Refraktärzeit läuft  
⇒ PVC ersetzt Sinussig.

15.05.2019 Erste Version des MC-Papers ist fertig

Zieljournal: npj Systems Biology and Applications

Outline:

Introduction: Probleme mit aktuellen Modellen (Reproducierbarkeit), Einordnung in macro/micro-/multilevel, challenges for multi-level modelling, Requirements (modular, human-readable, open-source, descriptive, hybrid, graphical)

Results: SHM monolithic, modular contraction, comparison, PVC use case + Integration + plot

Discussion: sensitivity of differences, plausibility of PVC model, benefits of requirements

Methods: software, code for MC + PVC, experiment setups, reference to gitlab

Supplement: language evaluation, full code

Research questions:

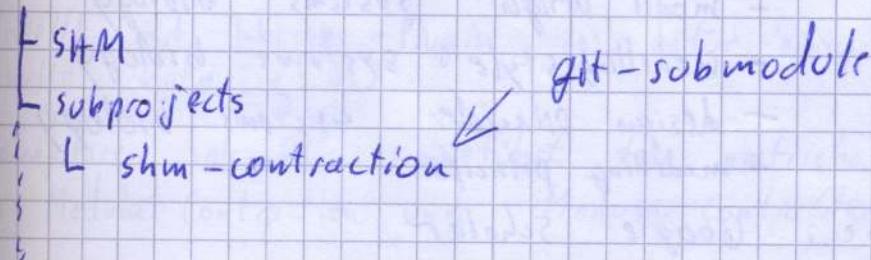
RQ1: What features are needed for a modeling language for systems biology?

RQ2: Can choice of language affect usefulness of a model?

Auslagerung von modular-contraction nach Github 21.05.2019

URL : <https://github.com/CSchoell/shm-contraction>

## Struktur in altem STM-Repo:



Code von shin-contraction gesäubert und dokumentiert 22.05.2019

- docstrings für Parameter und Variablen
  - SI-Funktions verwendet, wo es möglich war
  - Typ "Instant Signal" für Signale mit Kronecker-Delta-Semantik
  - $t \in \{0, 1\}$  darf nicht als delaytes Signal erkannt werden
  - pacemaker darf sich nur resetten, wenn er selbst ein "autonomous" Signal generiert hat, nicht bei jedem Ort
  - ~~delay~~ delay ignoriert Signale von außen explizit während noch ein Signal "on hold" ist.
  - Variable  $T$  wird in Modular Contraction X integr. (wie bei normaler Modular Contraction)
  - macht Variablen explizit "discrete", wo möglich

## Kritik am MC-Paper von A. Goessmann

11.06.2019

- Aussage schärfen.
  - Requirements sind Result, nicht Introduction
  - Recherchieren, was Literatur zu best practices sagt
  - Rotter Faden: Wir hatten Probleme, wir haben Reg. aufgestellt, die haben die Probl. gelöst.

Suchbegriffe : - best practices systems biology  
 - model quality systems biology  
 - model design systems biology  
 - modelling style systems biology  
 - design principles systems biology  
 - modelling principles

Suchmaschine : Google Scholar

Resultate :

- Mit "best practice" und "model quality" ist meist nur biologische Qualität der Aussagen gemeint.
  - Nur ein Paper gibt Anweisungen, wie der Code aussehen soll
    - Stichwort "model engineering" → gibt aber keine weiteren Treffer unter diesem Begr.
  - Zu sprachen tauchen "open-source", "graphical", "~~descriptive~~" "descriptive" und "hybrid" als Requirements auf, werden aber nie explizit besprochen
  - COMBINE listet Standardsprachen, MIRIAM legt Annotationsstandards fest, aber das war es dann doch an Standards
- ⇒ Was wir machen ist wirklich neu!  
 (zwischen Biologie und reinen Coding-Guidelines)

23.07.2019

Definition v. klaren Startwerten in shin-contracture

start = x ist nur ein Default-Wert in Modelica, ~~wenn~~ der von Tools auch komplett ignoriert werden kann. Fixed = true fügt eine initial equation hinzu, die man aber auch selbst explizit angeben könnte.

Richtlinie:

- fixed = true wird bevorzugt
- initial equation nur wenn Variable im Elternmodell definiert ist

Grafische Annotationen und Connektoren für shm-conctr. 10.8.2019

Icons in eigenem Paket, eingebunden mit extends:  
Gate, Heart, Hourglass, Metronome

Erstellt mit Inkscape-Plugin, das dafür einige Verbesserungen erhalten hat. (s.u.)

Konnektoren als Alias eingeführt für grafische Verbindungen in Modular Contraction und Modular Contraction X

bsp.: connector instant Input = input instantSignal;

Normalization, Float rounding, idk better String handling für Inkscape 10.8.2019

Mehrere Verbesserungen waren nötig, um die Icons für shm-contraction zu übersetzen:

Normalization: OptiModelica nimmt für icons immer ein Standard-Koordinatensystem mit Extent {100, 100, 100, 100}. Außerdem werden Linien sehr viel dicker angezeigt.  
⇒ Normalisierung der Werte aus dem SVO. (bei Beachtung der affinen Transformationen)

Float rounding: sowohl beim Lesen als auch beim Schreiben von Floats gab es Probleme. Lesen von scientific notation ist jetzt möglich und Modelica-Dateien enthalten höchstens Floats mit 3 Nachkommastellen.

String handling: Weil wir repräsentieren müssen, müssen wir auch mit "..." und "..." umgehen können.

SVG Paths: Support für horizontal und vertical line to wurde hinzugefügt.

Umbenennung von shm-contraction in shm-conduction 25.9.2019

"Contraction model" klingt nach finite-Elemente-Simulation der Ventrikel. Eigentlich simulieren wir aber das "cardiac conduction system".

⇒ Alle (missverständlichen) Vorkommnisse von "contraction" im Code wurden durch "conduction" ersetzt.

## Julia-Script für Test mit Travis CI zu shan-conduction hin

Mit OMJulia kann man auf die ZMQ-API von OpenModelica zugreifen. Damit habe ich ein Script mit folgenden UnitTests geschrieben:

### Simulate examples / UnidirectionalConduction Example

- loadModel(..) gibt "true" zurück
- loadModel(..) erzeugt keine Fehler (getErrorString() leer)
- simulate(..) erzeugt keine Fehler (---||---

### Simulate examples / PVC Example

- gleiche Tests wie oben für PVCExample
- und für Dummy-Modell PVCNoSinus

```
model PVCNoSinus
  extends SHAMConduction.Examples.PVCExample C
  with_Sinus = false
end PVCNoSinus;
```

(strukturelle Parameter lassen sich nicht überschreiben)

Besonderheit: Script hängt sich manchmal auf, weil nach dem senden von "quit()" an den OMC keine Antwort folgt.

## Dritter Draft der MC-Papers

- Supplement fertiggestellt mit Language selection, Diagramm des monolithischen Modells und vollständigem Code der Modelle
- Code im Paper aktualisiert
- Feedback der Coautorin eingearbeitet

5.10.2019

Beginn der Arbeit an HTT-modelica (Modularer Hodgkin-Huxley) Erst

Idee: Bevor ich mich auf Iuada 2009 stütze, wende ich die Guidelines aus dem MC-Paper auf das originale Hodgkin-Huxley-Modell an, da Iuada auf HTT-Gleichung basiert.

18.10.2019

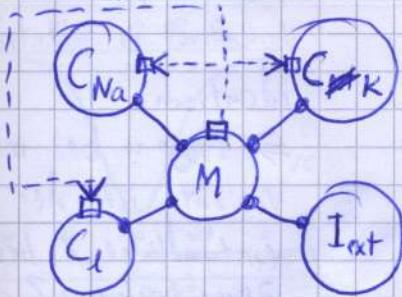
Einreichung des MC-Papers bei NPJ-SBA

Journal: Nature Partner Journals - Systems Biology and Applications

Diverse Statements hinzugefügt, Figures aktualisiert, Verweis auf Supplement angepasst, Repository veröffentlicht

# Fertigstellung des modularen Hodgkin-Huxley - Modells

22. 10. 2019



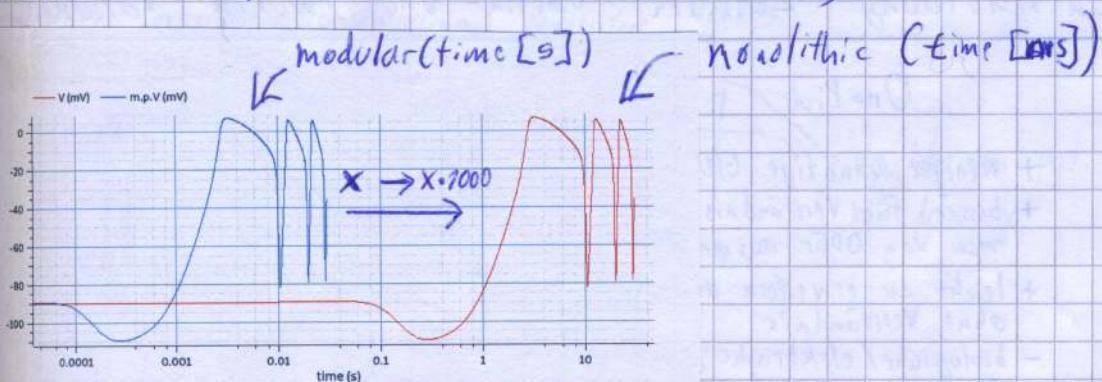
--- Temperatur  
— elektr. Verbindung  
 $C_{Na}$  = Natriumkanal  
 $C_K$  = Kaliumkanal  
 $C_L$  = Leck-Kanal  
 $I_{ext}$  = externe Stromquelle  
replaceable

Fitting functions sind als "function" realisiert, die bei der Redeklaration default-Werte für inputs zugewiesen bekommen.

Bsp.: redeclare function  $f_{beta} = \text{scaledExpF}(sx=1, sy=2)$ ;

Wiederverwendung : Gate-Modell (3x), Basisklasse Ionchannel (3x),  
Fitting functions (2-3x)

Resultierender Verhalten ist absolut identisch zu monolithischem Modell (bis auf angepasste Zeitskala, da Modelica immer Sekunden und nicht Millisekunden verwendet).



Reaktion

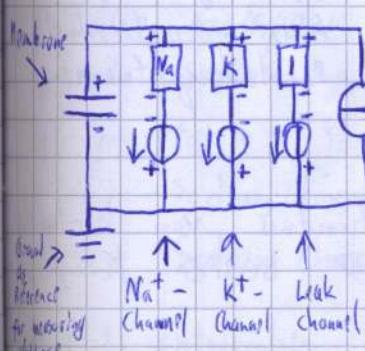
Promotion / results/  
HHmodelica /  
monolithic\_vs\_modular  
1p.1e8  
monolithic.csv  
modular 1p.csv

## Erste funktionierende TwoPin - Variante von HH-modelica

29. 10. 2019

Frage : Ist die Variante mit einem einzigen Pin realistisch, oder vereinfacht sie die Analogie zu sehr?

Idee : TwoPin - Variante, die den tatsächlichen Schaltkreis abbildet.



Erkenntnis: Viele Quellen zeichnen externe Stromquellen und/oder Ground nicht ein. Beide sind aber für eine sinnvolle Simulation nötig.

Promotion/results/  
Httmodelica/  
modular 1p-vs-modular  
2p.png  
modular 2p.csv



Ergebnis der Simulation:

Verhalten ist tatsächlich identisch.

⇒ Frage: bei welcher Variante ist der Code verständlicher / besser erweiterbar?

30.10.2019

Neues Framing des Promotionsprojektes: Modelica als Compiler für ODEs

Grace Hopper: Schreibe nicht direkt Maschinencode, sondern entwickle menschenverständliche Sprachen, die automatisiert in Maschinencode übersetzt werden können.

Christopher Schönleff: Schreibe nicht direkt mathematische Gleichungssysteme, sondern entwickle menschenverständliche Sprachen, die automatisiert in Gleichungssystem übersetzt werden können.

1.11.2019

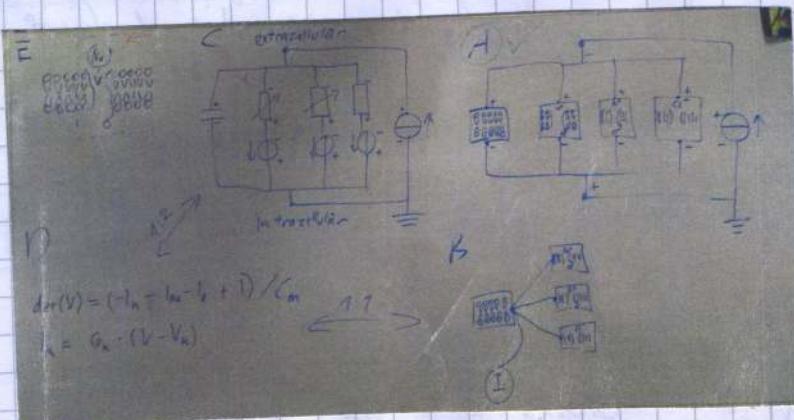
Entscheidung zwischen OnePin- und TwoPin-Variante

### OnePin

- + weniger unnötige Gleichungen
- + besser für Verständnis, wenn man von ODEs ausgeht
- + leicht zu erweitern auch ohne Verständnis
- biologische / elektrische Analogie geht verloren
- + bracht (fast) kein elekt. Verständnis

### TwoPin

- + verbindet biol. und elektr. Analogie
- + besser für Verständnis, wenn man von Biologie ausgeht
- + leichter zu erweitern als rein elektrisches Modell (Richtungen)
- + lässt sich im editor Schaltkreis einspielen (Voltage Clamp)
- + klare Trennung zw. intra- und extrazellulär
- braucht ein Minimum an elekt. Verständnis



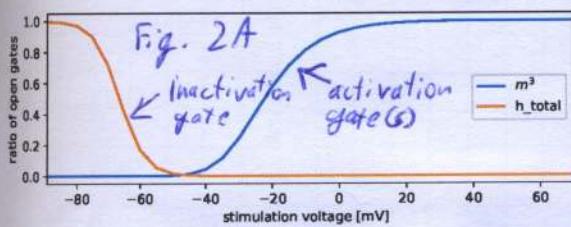
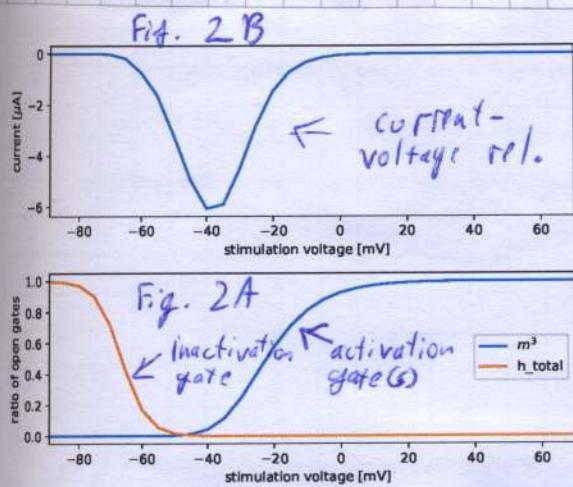
Erkenntnis:

Je nach dem eigenen Hintergrund bevorzugen unterschiedl. Menschen unterschiedl. Darstellungsformen.

⇒ TwoPin-Modell erlaubt sowohl Verständnis von biologischem als auch von technischer Seite ⇒ bester Komprom.

Inoda - Modell : Nat - Kanal reproduziert Lindblad 1997

6. 11. 2019



↑ Reproduktion von ~~Lindblad~~

Figure 2A und 2B aus  
Lindblad 1997.

Implementierung der  
Na<sup>+</sup>-Kanäle, die in  
Inoda 2009 verwendet  
werden (und aus  
Lindblad 1997 übernommen  
wurden).

Im Gegensatz zu den  
Hodgkin-Huxley-Kanälen  
basieren hier die Gleichungen  
auf der GHK-Flux equation  
(nicht zu verwechseln mit  
der GHK voltage equation).

GHK-flux equation für ein einzelnes Ion X:

$$I_{\text{max}} = P_x \cdot [X]_{\text{ex}} \cdot \frac{F^2}{RT} \cdot Z_x^2 \quad \leftarrow \text{maximale conductance}$$

↓ ↓ ↓ ↓

Permeability extracellular concentration temperature valence

$$I_{\text{open}} = g_{\text{max}} \cdot V \cdot \frac{e^{(V-V_{\text{eq}})} \frac{E}{RT} \cdot Z}{e^{V \cdot \frac{F}{RT} \cdot Z} - 1} \quad \leftarrow \begin{array}{l} \text{equilibrium pot.} \\ \text{von Ion X} \end{array}$$

↑

current when  
channel is fully open

Herleitung: (17) in Goldman 1943

- factor out ~~λ~~ λ<sub>+</sub>
- substitute  $\frac{\lambda_-}{\lambda_+} = e^{\beta V_0}$

InaMo: Aufteilung von SodiumchannelExample in IV und steady

12. 11. 2019

Fig 2B von Lindblad 1997 habe ich falsch interpretiert.

In diesem Szenario müssen kurze "test pulses"  
in der Zielspannung produziert werden, ~~und~~ keine durch-  
gehende Spannung dagelegt werden.

Als Resultat: Sodium Channel Example aufgeteilt in

- Sodium Channel ~~Steady~~ Steady (gleiches setup wie zuvor)
- Sodium Channel IV (test pulses mit entsprechendem Hilfsmodell)

\* Plots entsprechen leider noch nicht der Referenz.

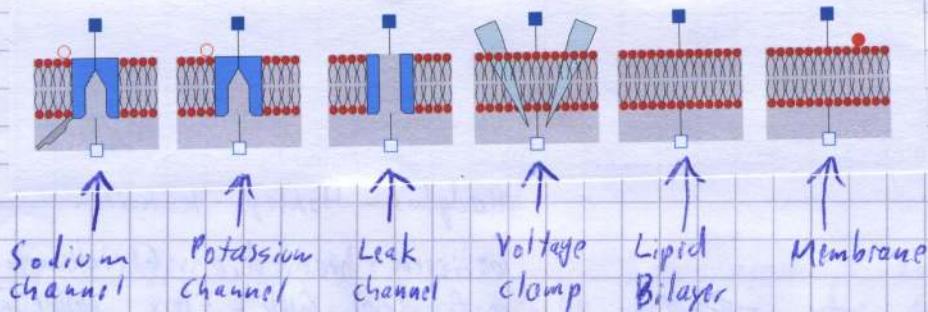
results / InaMo/  
Sodium channel Example  
.csv, .pdf

InaMo / Examples /  
Sodium Channel Example  
.mo

71

13.12.2019 HH-modelica: Icons und neue Dateistruktur

Als Vorbereitung zur Publikation hat HH-modelica jetzt Icons für alle relevanten toplevel-komponenten und die Dateistruktur wurde angepasst, um die TwoPin-Variante mehr in den vordergrund zu rücken.



19.12.2019 ~~HH~~ HH-modelica: Erster Draft des Papers fertig

Paper / eigenes /  
2020-HH-modelica

RQ1: Can the understandability of the HH-model be improved by a modular implementation that bridges the gap between biological meaning and electrical analogy?

RQ2: Can a modular implementation of the HH-model serve as a unifying basis for extensions and therefore facilitate the creation of more complex HH-type Models?

Unterstützung: cognitive load theory als Modell für ~~Verständlichkeit~~ "verständlichkeit".

HH-modelica ist seit 13.12. öffentlich auf GitHub zu finden (presub-inquiry).

30.12.2019 NC-Paper: Feedback von Reviewern

Wtoit: Major revision

Reviewer #1:

- bemängelt fehlende Referenz zu Python-tools und anderen state-of-the-art-Ausätzern
- glaubt wir würden behaupten, Modelica wäre besser als alles andere
- wünscht sich eigentlich ein anderes paper (declarative vs. imperative Meta-Study)

## Reviewer #2:

- bemühtet, dass wichtige Charakteristika nicht aufgeführt wurden (ontologisch, solvers, ...)
- einige Unklarheiten im paper (style vs grammar, was meint "open-source")
- SHM sei kein typisches Modell und nur begrenzt reusable

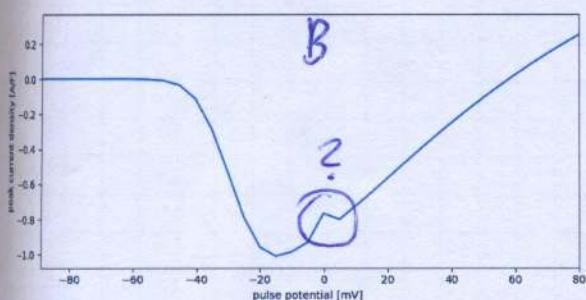
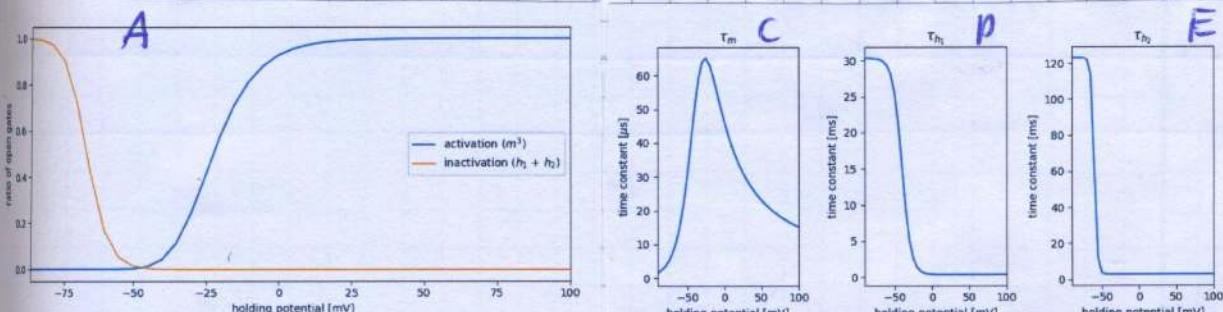
MC-Paper: ~~Erster~~ Erster Draft der Revision ist fertig 30.01.2020

Strategie:

- Titel und Abstract "entschärft"
- Mehr Hinweise, dass Modelica nur eine von vielen Lösungen ist
- Abschnitte zu state-of-the-art-Sprachen in Introduction und spezifisch zu Python im Supplement
- Erweiterung der ~~kurzen~~ Beschreibung der Charakteristika in results und in discussion um die von Reviewer #2 vermissten und von Reviewer #1 missverstandenen Themen

InMo: Reproduktion von Lindblad 1997 gelückt!

7.02.2020



Beobachte Fehler:

- Vorzeichenfehler in Ber. des Equilibriumpotentials
- Copy-paste-fehler in der Berechnung von tau\_m

results/InoMo/  
lindblad 1997\*.pdf/png

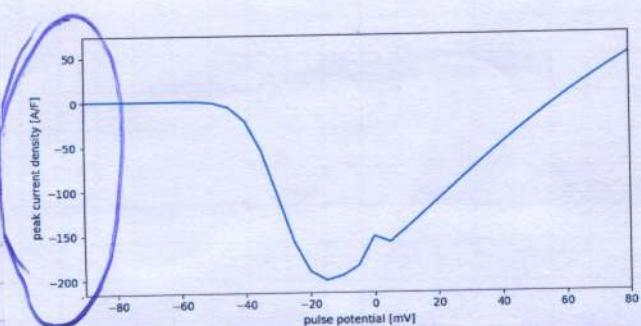
Offene Fragen:

results/InoMo/  
sodiumchannel\*.csv  
- nernstfix.csv

- Warum ist "gemessenes"  $\tau_m$  ziemlich so groß wie erwartetes?
- Was hat löst den Fehler in Fig. B aus?
- Warum stimmen Werte für Fig. B nicht (anders C?)?

11.02.2020 Ina Mo: Sodium channel IV hat korrekte Größenordnung Ina Mo

results/InaMo/  
Lindblad 1997B  
-oomfix.png



Änderungen:

- Kapazität auf  $50 \mu F$
- Permeabilität im gbkFlux ist keine SI-Einheit  $\Rightarrow$  Definition für Permeability FM hinzugefügt (Einheit  $\frac{m^3}{s \cdot m^2} = \frac{m}{s}$ )

results/InaMo/  
sodium channel IV  
-oomfix.csv

- gbkFlux gibt jetzt CurrentDensity an, nicht Current. (wird im Ion channel GTHK mit unit area multipliziert, um Einheiten korrekt zu halten)
- setzt sodium.p auf  $1.4 \cdot 10^{-15}$  statt  ~~$1.4 \cdot 10^{-12}$~~ , wobei auch mit korrekten Einheiten  $0.0074 \frac{\mu A}{s}$  eigentlich  $1.4 \cdot 10^{-12} \frac{m^3}{s}$  wären. Inada verwendet nämlich auch  $1.4 \text{ pA/s}$  statt  $1.4 \text{ nA/s}$ , und dieser Wert liefert realistische Ströme im nA-Bereich (und korrekten Plot, s.o.)
- peak-indicator-Variablen ( $denc(i) < 0$ ) wird verwendet statt instabilem Ausdruck  $x = \max(prec(x), y)$ .

Kleine Inkonsistenz am Rande: Colatsky 1980 gibt für  $P_{Na}$  den Wert  $10^{-5} \frac{cm}{s} = 10^{-7} \frac{m}{s}$  an. ~~Das passt~~

(auch für Zellen im Atrium des Kaninchenerzens). Das passt überhaupt nicht zu unseren  $10^{-15} \frac{m}{s}$ , aber ich behalte die Zahl trotzdem bei, um Lindblad und Inada zu reproduzieren.)

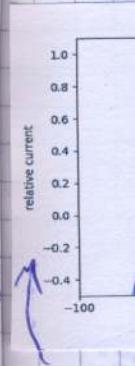
14.02.2020 HH-modellica: Zweiter Draft des Papers fertig

Hauptsächliche Änderungen:

- CLT besser mit Rest verknüpft
- Bessere Motivation durch Erklären möglicher Probleme
- Redundanzen entfernt
- Bessere Argumentation zu RQ1

Zieljournal: IEEE Transactions on Biomedical Engineering

Offene Fragen: Gernote input zu HH-type Modellen mit leichten Erweiterungen fehlen noch.



geteilt von  
Außen

MC-Pl

LaTe  
-  
-  
-

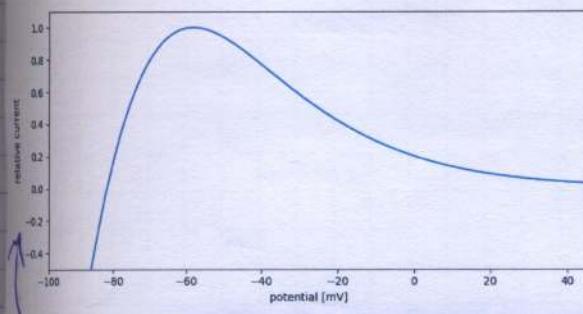
Ina Mo  
Icar  
richt  
Plots

Ina Mo  
Inca  
aber  
Beson

- Inca  
Die  
Diffe  
close  
proc  
Evtl

## InMo: Inward Rectifier implementiert und verifiziert

16.02.2020



geteilt durch max. Stromstärke von  $\sim 77 \text{ pA}$

Außerdem: Idee wo Fehler in Fig 2B herkommt:  
Das Phänomen tritt bei  $V=0$  auf  
 $\Rightarrow$  vielleicht Fehler in Vermeidung der Unsymmetrie des Zellzyklus im glik Flux?

MC-Paper: Revision eingereicht

17.02.2020

LaTeX-Datei mit script bereinigt zum Einreichen

- bibfile in Datei mit filecontents integriert
- Bilddateien aus unterordner in Hauptverz. verschoben
- Kommentare gelöscht

## InMo: Implementierung der meisten Ionenkanäle ohne Verifikation 18.02.2020

$I_{CaL}$ ,  $I_{to}$ ,  $I_{Kr}$ ,  $I_F$  und  $I_S$  implementiert, aber noch nicht getestet. (Nur Check Model funktioniert schon.)

Plots zur Verifikation: Inada 2009 (Supplement) Fig S1-S5

## InMo: Implementierung der übrigen Kanäle und Pumpen

8.03.2020

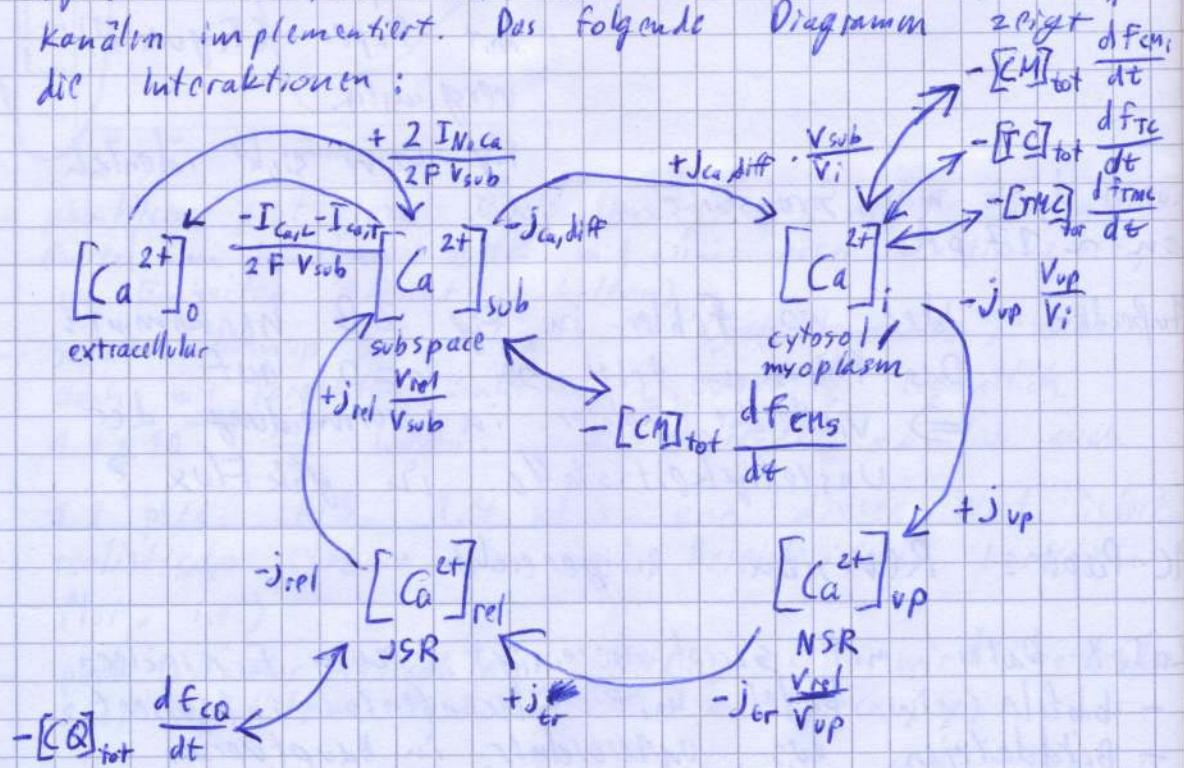
$I_{Na}$ ,  $I_P$ ,  $I_b$  und intrazelluläres  $\text{Ca}^{2+}$ -Handling implementiert, aber noch nicht getestet. (Check Model funktioniert)

Besonderheiten:

$I_{NaCa}$  besteht aus Modell mit 4 Zuständen.  
Die Zustandsübergänge sind nicht wie üblich als Differentialgleichungen gegeben, sondern als direkte Iosod-form solution mit einer "generalized King-Altman procedure for cyclical, unbranched state diagrams".  
Evtl. kann das Modell noch besser modularisiert werden,

wenn ich diese Methode besser verstehen.

- Das  $\text{Ca}^{2+}$ -Handlung habe ich aus den Formeln von Kurata 2002 und Dokos 1996 ~~zusammen~~ reverse-engineert. Die Formeln sind zum Teil in einem eigenen Modell, zum Teil aber auch in den  $\text{Ca}^{2+}$  transporthingkanälen implementiert. Das folgende Diagramm zeigt die Interaktionen:



Die Übergänge zwischen den Kompartimenten sind wegen der Volumenverhältnisse nicht symmetrisch. Den Grund dafür verstehe ich noch nicht. Insbesondere unterscheiden sich hier die Formulierungen bei Kurata 2002 und Dokos 1996:

$$\frac{d [Ca]^{2+}_i}{dt} = -j_{up} \frac{V_{\text{up}}}{V_i} + \dots$$

$$\frac{d [Ca]^{2+}_i}{dt} = -\frac{j_{up}}{V_i} + \dots$$

$$\frac{d [Ca]^{2+}_{\text{up}}}{dt} = j_{up} + \dots$$

$$\frac{d [Ca]^{2+}_{\text{up}}}{dt} = \frac{j_{up}}{V_{\text{up}}} + \dots$$

Kurata 2002

???

Dokos 1996

9.03.2020 InaMo: Bugfix für seltsames Verhalten von  $I_{Na}$  bei  $V \approx 0$

Erste Idee: Problem tritt bei  $V=0$  auf  $\Rightarrow$  Formel für glik Flux bei  $V=0$  hat einen Fehler.

Diagnose: Der Formel fehlte zwar ein  $Z$ , aber das war nicht der Grund für das Fehlverhalten. Bei einer Veränderung der "Schrittwelle" in der die Pulse-Spannung erhöht wird trat der Fehler entweder gar nicht auf, oder die Simulation brach mit einer Division durch 0 ab.

$\Rightarrow$  Grund ist, dass nur abgefragt wird, ob  $V \approx 0$  ist. Bei  $V \approx 0$  entstehen durch Rundungsfehler

Astefakte, weil nahe der Unstetigkeitsstelle die Fehlerfortpflanzung sehr ungünstig ist.

Lösung: Abfrage geändert auf "abs(v) < 1e-6" statt "v == 0".

Resultat:

