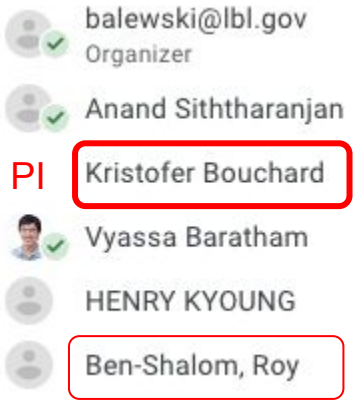
Characterization of neuron physiology by regression of its time-dependent response with CNN

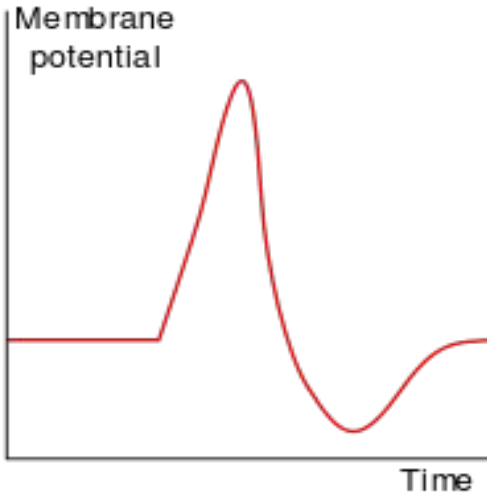
Target audience : physicists & ML experts
Summary: work in progress, new problems as we drill deeper - very exciting !!!

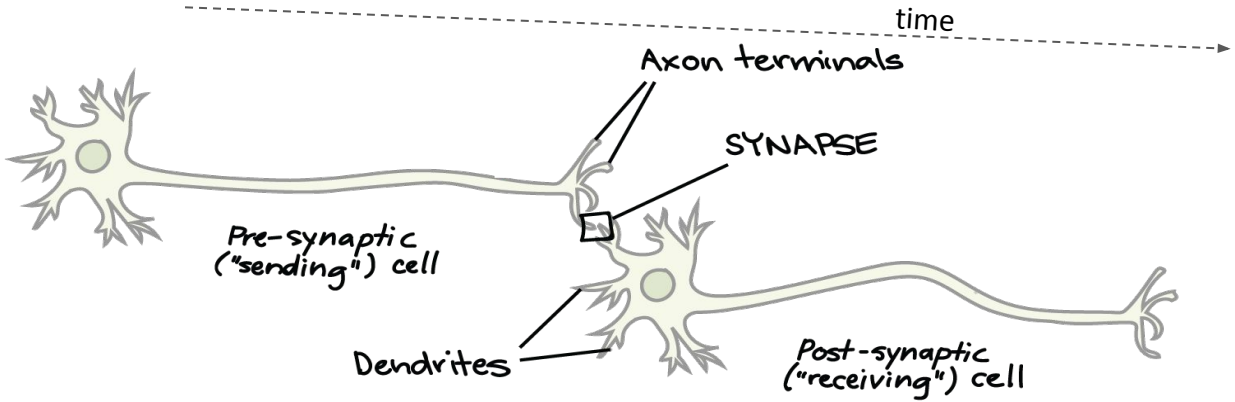
- What is a neuron?
- Neuron simulators/ training data set
- ML objective
- Designed ML model #1
- Random Hyper-param scan
- ML model #693 - hard case predictions
- ‘Adversarial’ predicting
- Living cell measurement and analysis
- Outlook

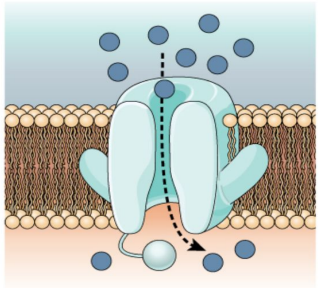
Collaboration with



Ion channels control electrical properties of neuron







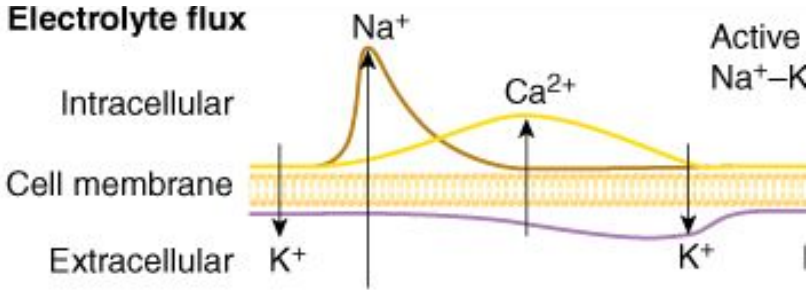
Open In response to a nerve impulse, the gate opens and Na^+ enters the cell.

Sodium
Potassium
Calcium

Why understanding ion channels is important ?

- There is a need for experimental techniques to measure distribution of ion channels, it will help us understand better how neurons work.

- Channelopathies
  - Epilepsy
  - Autism



- Accurate neuron models can help in find targets for treatments
- Build better neuronal networks that simulate neuronal circuits

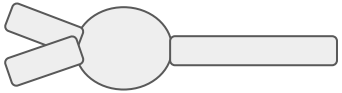
Neuron response simulators

a) **Izhikevich**: point cell

$$v' = 0.04v^2 + 5v + 140 - u + I$$
$$u' = a(bv - u)$$

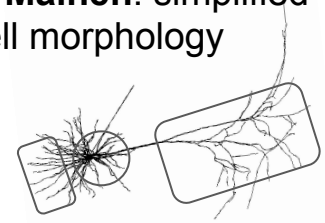
if $v = 30$ mV,
then $v \leftarrow c$, $u \leftarrow u + d$

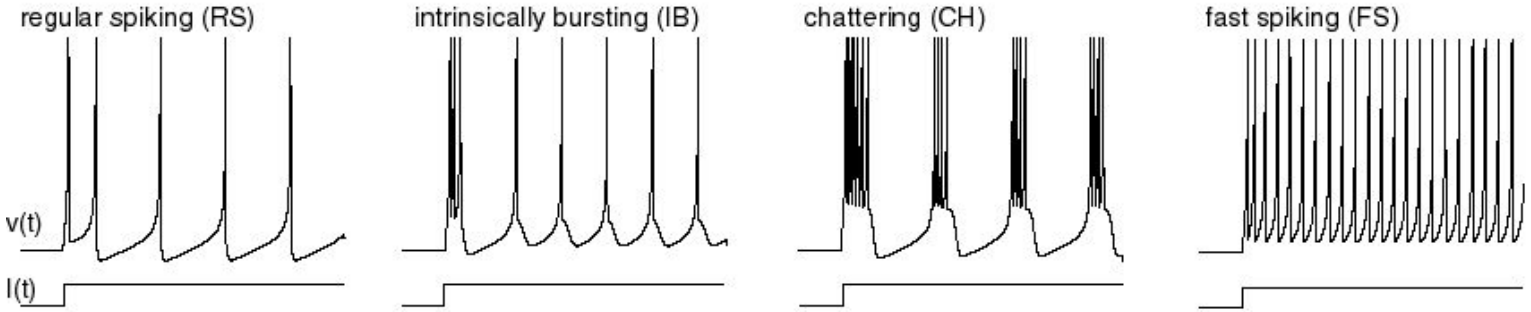
b) **Hodgkin–Huxley**: ball with 2 sticks cell geometry



$$I = C_m \frac{dV_m}{dt} + g_K(V_m - V_K) + g_{Na}(V_m - V_{Na}) + g_l(V_m - V_l)$$

c) **Mainen**: simplified cell morphology



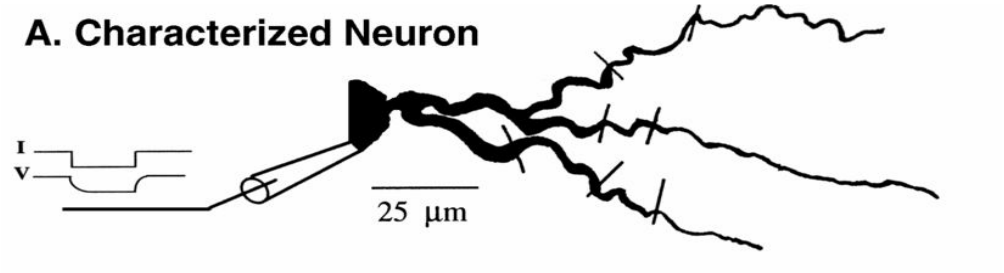


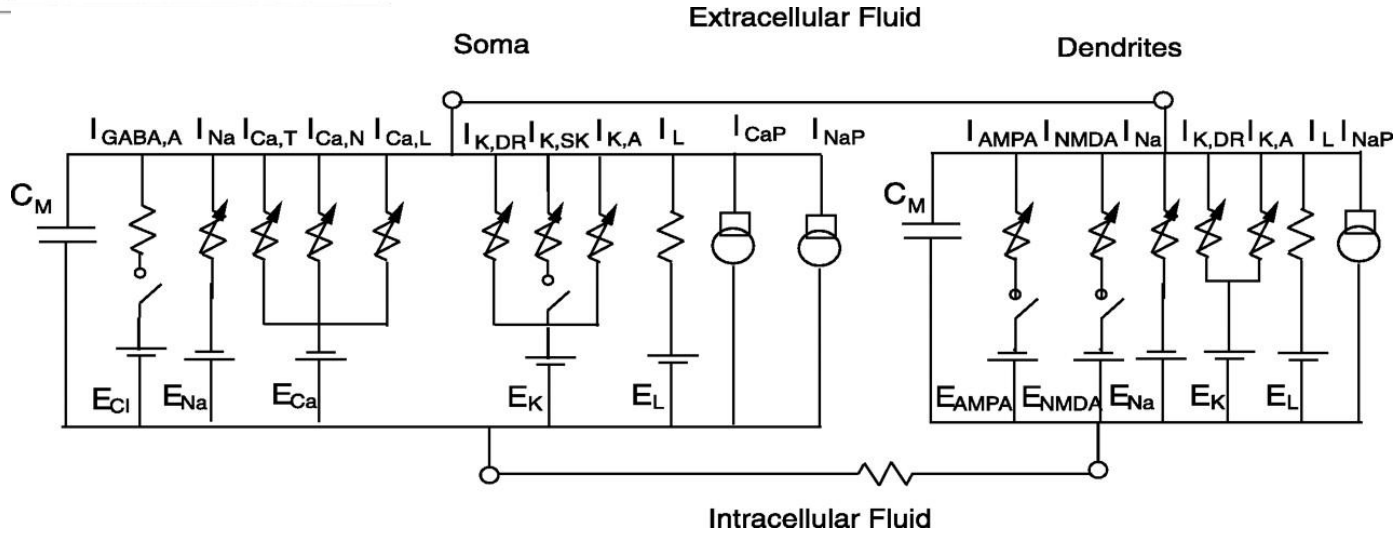
Compartmental cell model (Mainen)

Influence of dendritic structure on firing pattern in model neocortical neurons

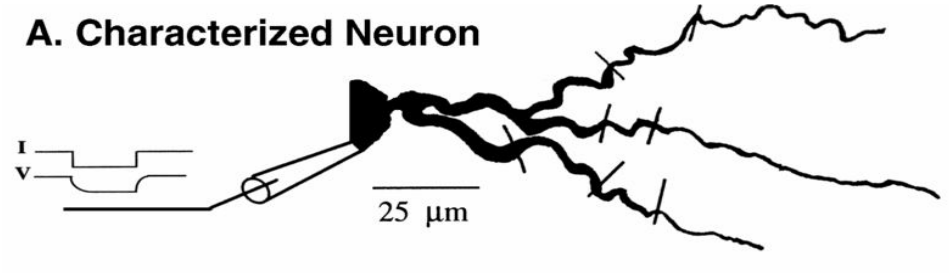
Zachary F. Mainen* & Terrence J. Sejnowski

Howard Hughes Medical Institute, Computational Neurobiology Laboratory, Salk Institute for Biological Studies, La Jolla, California 92037, and Department of Biology, University of California, San Diego, La Jolla, California 92093, USA

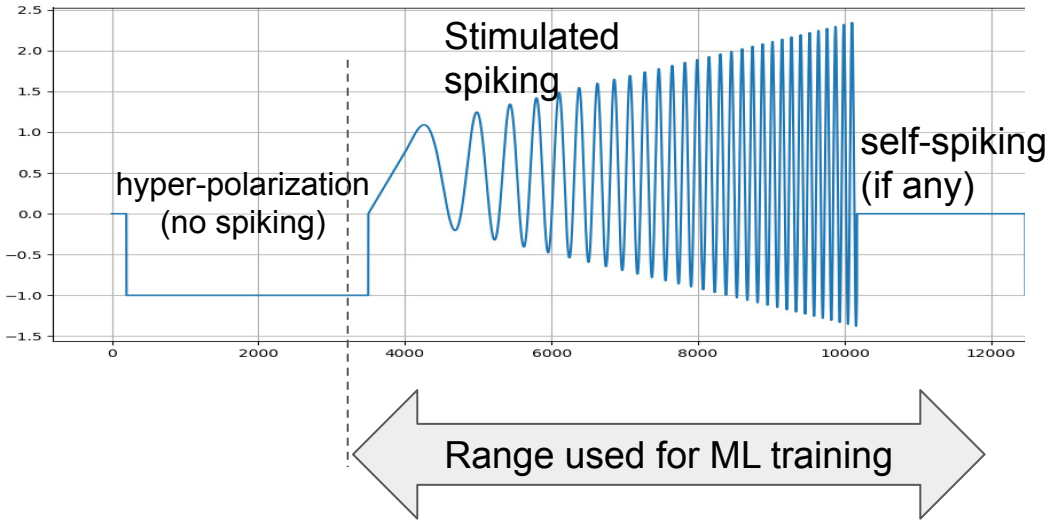




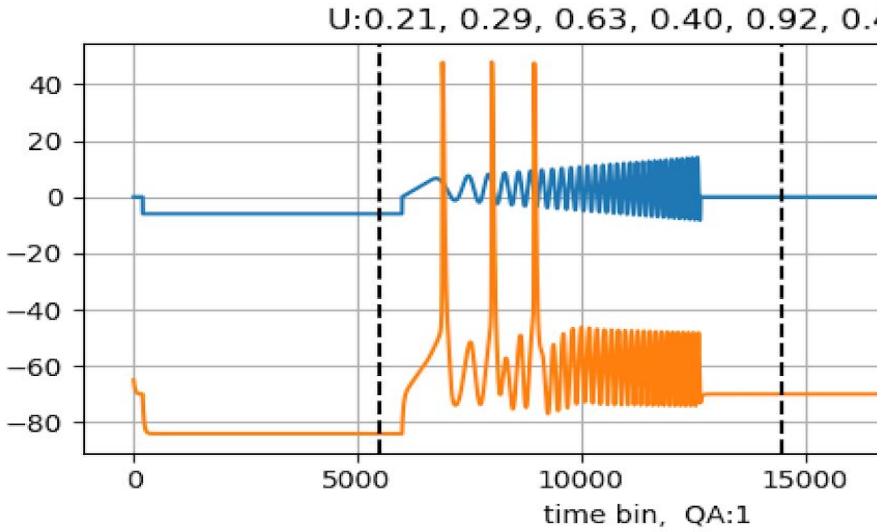
Stimulus: ‘reset’+chirp

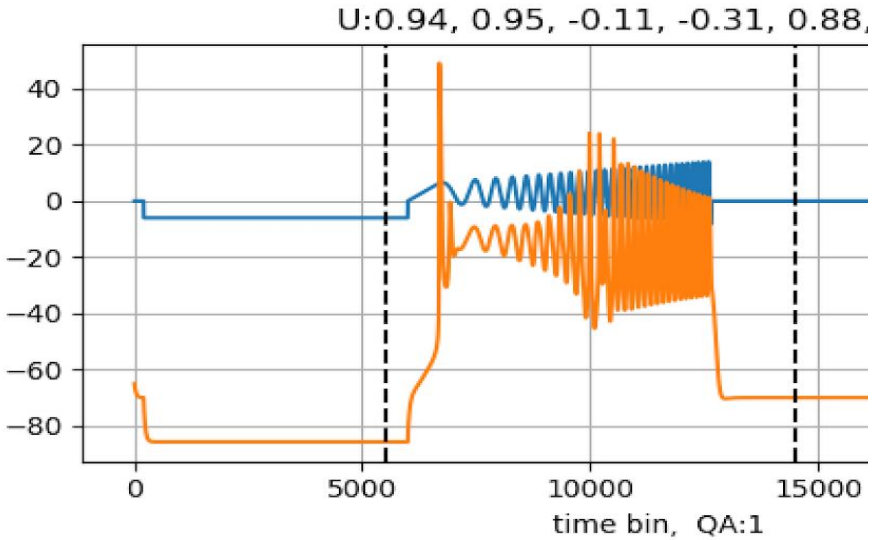


INPUT



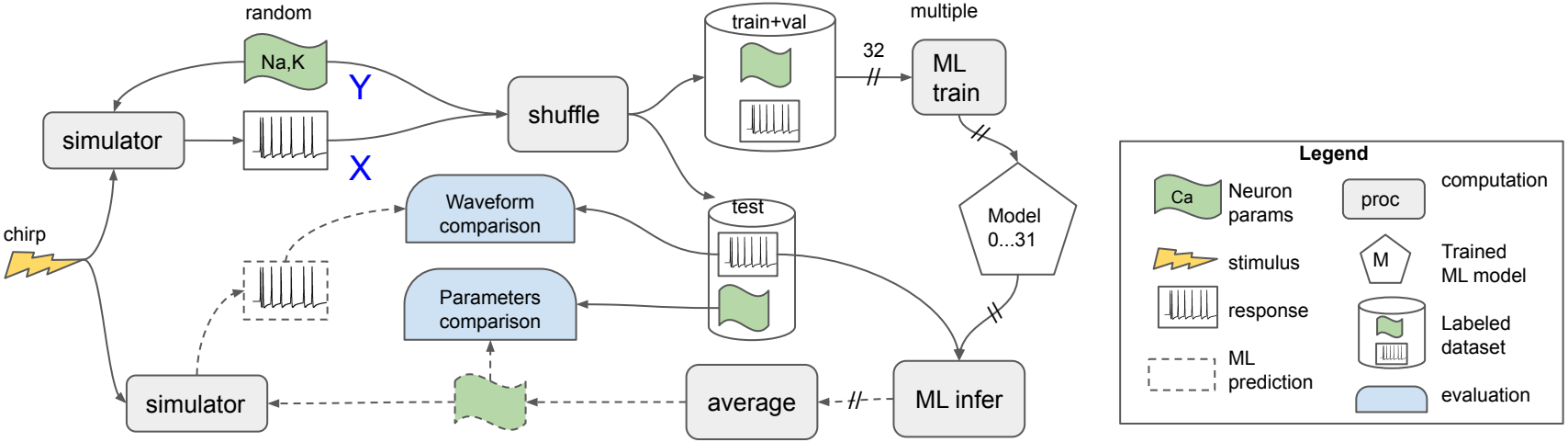
Cell response to stimulus depends on its properties





ML training objectives

1. Map 1D time series of cell response to handful of ion channel conductance params describing the cell properties (in a framework of specific cell model)
2. Provide error for predicted params (by training an ensemble)
3. Provide ‘out-of-range’ warning for a-typical time series (e.g. experimental data can be corrupted)



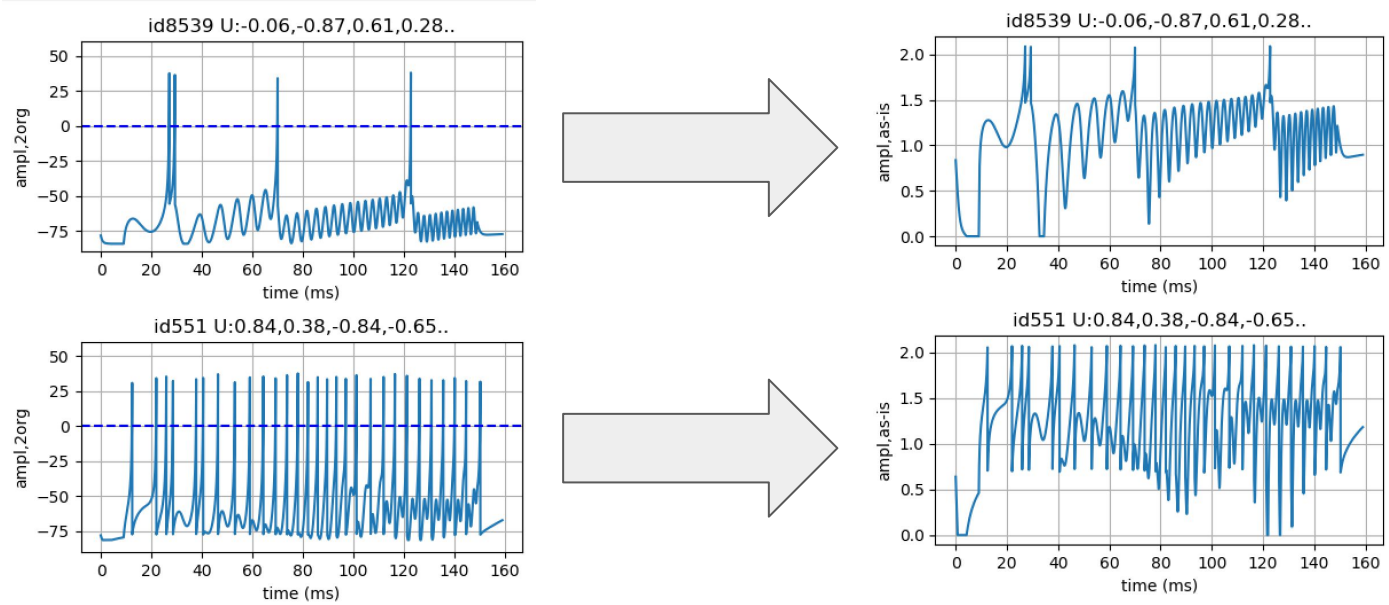
Data curation (X)

Traces (time dependent neuron response)

- raw range [-80,+50] mV
- Domain expert: amplitude of spikes has low information content
- Mapping: X'= log10(X+81/mV)

X original

X scaled

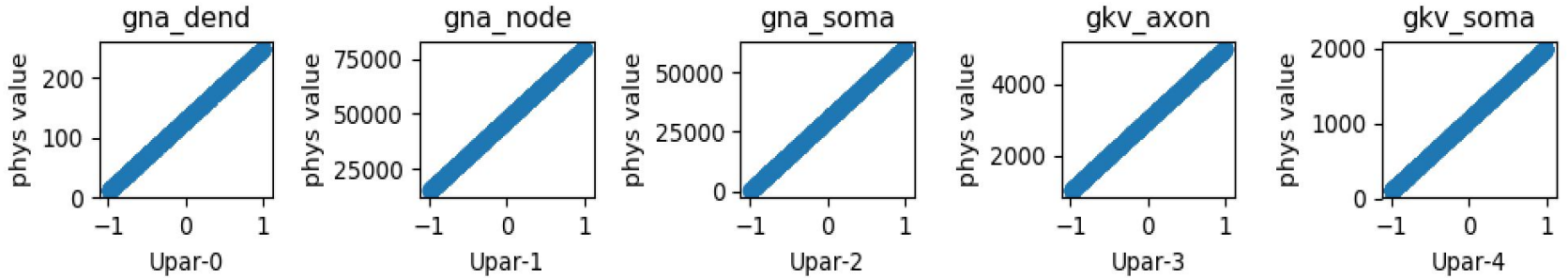


Data curation (Y)

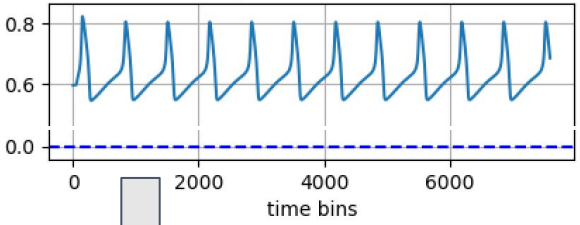
Cell params: Ion channels conductances , membrane resistance, m. capacitance, etc.

- raw range of phyPar is diverse
- Linear mapping: Y'= a+b*Y
- → unitPar range is [-1,1]

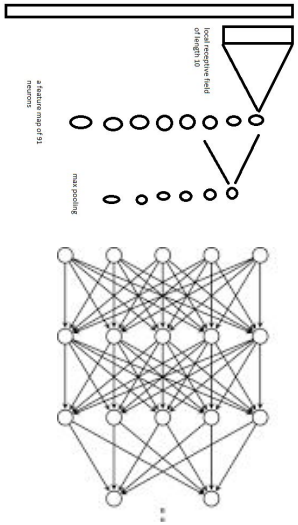
gna_dend	10	250	S/cm2
gna_node	15,000	80,000	S/cm2
gna_soma	10	60,000	S/cm2
gkv_axon	1,000	5,000	S/cm2
gkv_soma	100	2,000	S/cm2
gkm_dend	0.05	0.20	S/cm2
gkca_dend	1.5	6	S/cm2
gca_dend	0.15	0.60	S/cm2
c_m	0.3	1.5	uF/cm2
rm	15	60	kOhm*cm2



1D Convolutional Neural Network



Input: cell response
9000 floats, 1D vector
amplitude normalized



few CNN blocks
* local reception filed, kernel=5
* activation=LeakyReLU
* maxPool

Flatten

few FC blocks
* Dense(act=LeakyReLU)
* Dropout(0.01)

Output K floats: gna_dend
gna_node gna_soma ….
Dense(K, act=1.2*tanh)

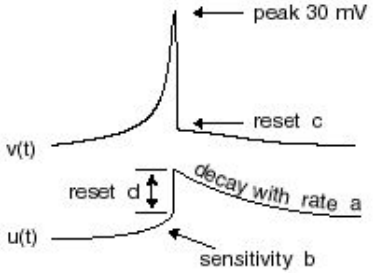
```
$ cat hpar_cellRegr_cnn1.yaml
# CNN params
conv_filter: [6,12,18,24,30,36]
conv_kernel: 5
conv_repeat: 2
pool_len: 3

# FC params
fc_dims: [20, 10, 10]
lastAct: tanh
outAmpl: 1.2
dropFrac: 0.02

# training
lossName: mse
optimizerName: adam
```

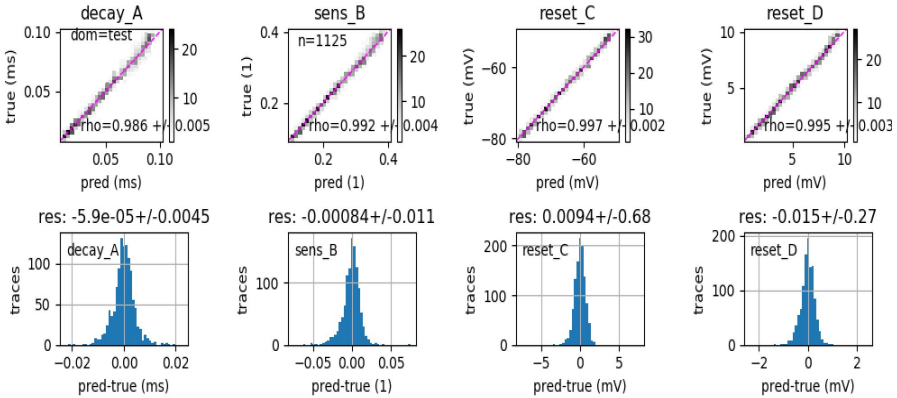
ML model #1 results

4-params Izhikevich neuron simulator

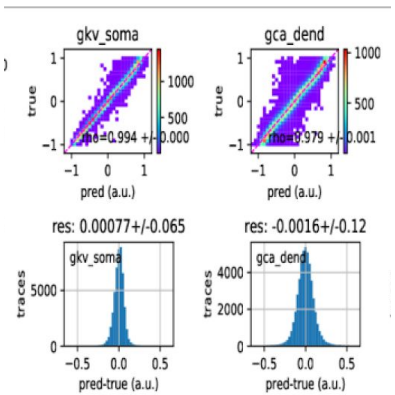
$$v' = 0.04v^2 + 5v + 140 - u + I$$
$$u' = a(bv - u)$$

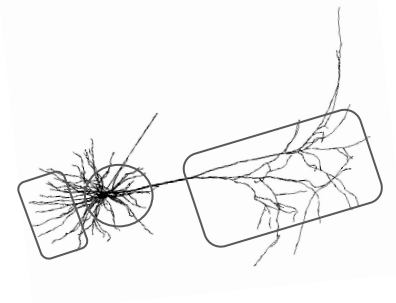
if $v = 30$ mV,
then $v \leftarrow c$, $u \leftarrow u + d$

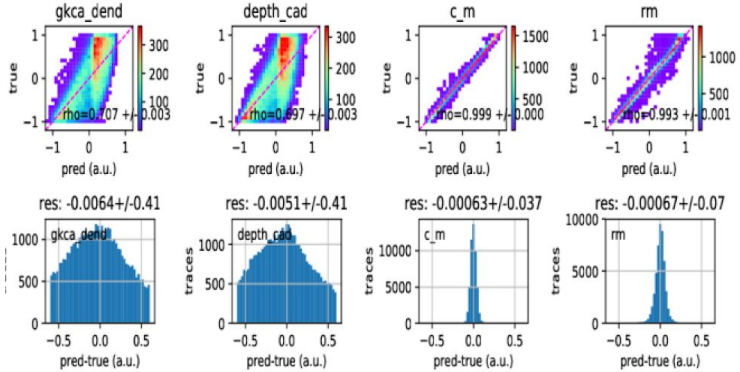




10 params Mainen 'morphological' simulator







Random search of ML hyper-parameters

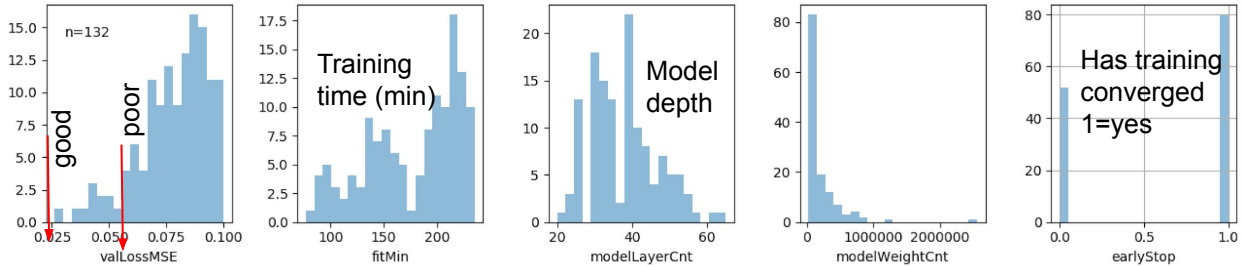
```
$grep np.random genHPar_CellHH.py
np.random.seed() #  set the seed to /dev/urandom
    numLyr=np.random.randint(2,8)
    filt0=np.random.randint(2,21)
    filtIncr=np.random.randint(0,8)
    kern=np.random.randint(3,8)
    pool=np.random.randint(3,8)
    repeat=np.random.randint(1,4)
    numLyr=np.random.randint(2,8)
    filtEnd=np.random.randint(nOut,51)
    filtIncr=np.random.randint(0,20)
    #lastAct=str(np.random.choice(['linear','tanh']))
    #ampl=np.random.uniform(1.0,2.0)
    dropFrac=float(np.random.choice([0.01, 0.02, 0.05]))
    #loss=str(np.random.choice(['mse','mae']))
    opt=str(np.random.choice(['adam','nadam''adadelta']))
    BS=1<<np.random.randint(4,8)
    lrReduce=np.random.uniform(0.2,0.8)^2

while True: # reject invalid models
    hpar1=get_CNN_HPar()
    if isValid_CNN_HPar(args.nInpFeat,hpar1) : break
```

$cat hpar_cellRegr_693.yaml

```
# CNN params
conv_filter: [18, 25]
conv_kernel: 5
conv_repeat: 3
pool_len: 7

# FC params
dropFrac: 0.01
fc_dims: [148, 131, 114, 97, 80, 63, 46]
lastAct: tanh
outAmpl: 1.2

# training
batch_size: 32
lossName: mae
optimizerName: adam
reduceLR_factor: 0.14
```

Hyper-param scan (checked 400+ models)

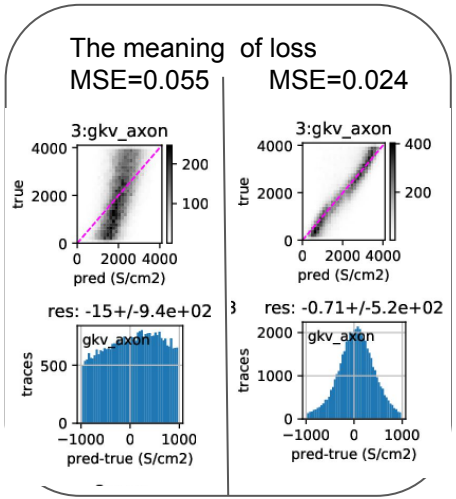
Loss-ranked list of 400+ scanned models, top 130 w/ loss<0.1 shown



Top-3 contenders: 71443_410, 71443_335, 71047_203
had initial losses of 0.026, 0.037, 0.040

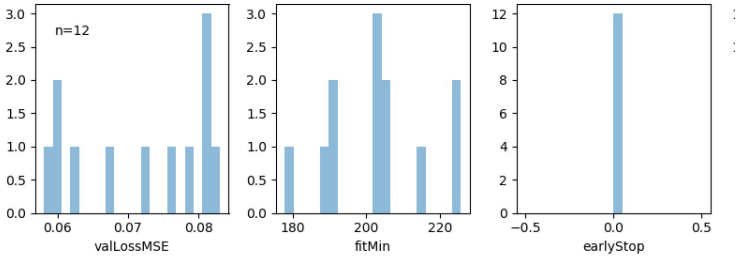
Golden-model 203

	design	valLoss MSE	valLoss	K off	loss Name	early Stop	epochs	fit(min)	optimizer	#param(k)	flat size	#layer	batch size	conv_filter	conv kernel	pool len	fc_dims	dropFrac	lastAct
2	71443_410	0.026	0.026	2	mse	0	49	214.1	adam	458.6	2750	58	256	(3, [19, 28, 37, 46, 55])	7	3	[105, 87, 69, 51, 33, 15]	0.01	tanh
3	71443_335	0.037	0.037	7	mse	1	62	185.4	nadam	288.6	6920	31	64	(2, [13, 22, 31, 40])	7	3	[36, 26]	0.01	linear
4	71047_203	0.04	0.04	3	mse	0	36	192.6	adam	106.4	460	42	16	(2, [18, 25, 32, 39, 46])	6	4	[80, 63, 46, 29]	0.02	tanh
5	71047_50	0.041	0.122	2	mae	0	47	219.4	nadam	734	6630	45	64	(3, [15, 23, 31, 39])	7	3	[99, 80, 61, 42]	0.02	linear
6	71047_196	0.041	0.121	4	mae	1	78	218.9	adam	110.7	539	42	128	(2, [17, 25, 33, 41, 49])	5	4	[82, 69, 56, 43]	0.05	linear
7	71047_212	0.043	0.116	4	mae	0	63	216.9	nadam	110.3	328	51	256	(3, [20, 27, 34, 41])	5	6	[101, 86, 71, 56, 41, 26]	0.01	linear
8	71443_246	0.045	0.046	6	mse	0	41	201	nadam	52.8	1449	32	16	(3, [9, 16, 23])	7	6	[25, 18]	0.01	linear
9	71443_437	0.045	0.045	5	mse	1	62	146.9	adam	92.9	784	44	64	(1, [14, 21, 28, 35, 42, 49])	7	3	[58, 51, 44, 37, 30, 23, 16	0.01	tanh
10	71443_242	0.049	0.054	2	mse	0	43	220.2	adam	486	6600	35	16	(3, [16, 23, 30])	7	4	[68, 52, 36]	0.01	tanh
11	71047_216	0.05	0.128	0	mae	1	68	166.1	nadam	78.9	1014	35	128	(3, [10, 18, 26])	6	7	[57, 41, 25]	0.01	tanh
12	71443_321	0.053	0.053	1	mse	1	70	139.3	nadam	126.8	1350	29	64	(1, [19, 21, 23, 25])	7	4	[77, 67, 57, 47]	0.02	tanh
13	71047_211	0.056	0.056	3	mse	0	52	221.5	nadam	79.3	1512	31	16	(2, [19, 22, 25, 28])	4	4	[41, 26]	0.1	linear
14	71047_239	0.057	0.148	7	mae	0	48	225.6	adam	163	800	54	32	(3, [19, 26, 33, 40])	4	5	[106, 96, 86, 76, 66, 56, 4	0.05	linear
15	71443_324	0.057	0.142	4	mae	0	67	228.4	adam	312	3534	41	64	(3, [17, 24, 31])	3	5	[80, 69, 58, 47, 36]	0.02	tanh

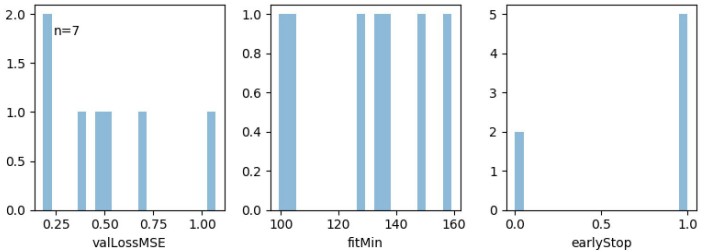


Stability of top 3 ML models

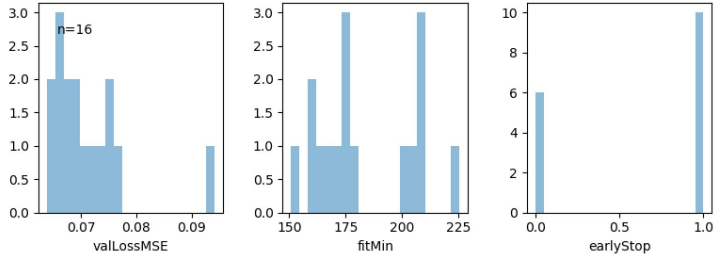
_410 : large spread of loss
4 of 16 timed out (diverged ?)



_335 : huge spread of loss
2 of 16 not converged yet (good!)

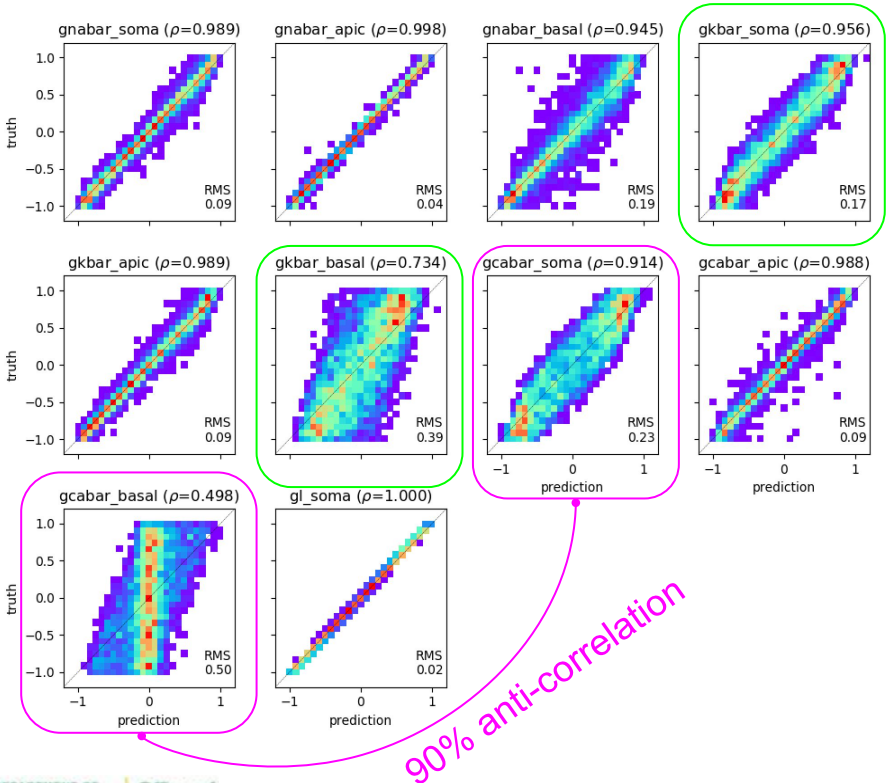


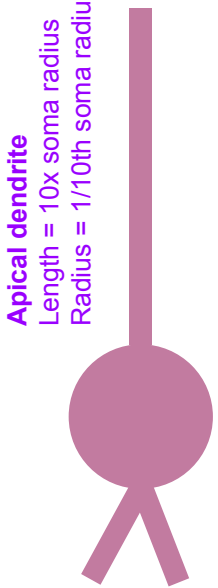
_203 : narrow spread of loss
6 of 16 not converged yet
(very good!)

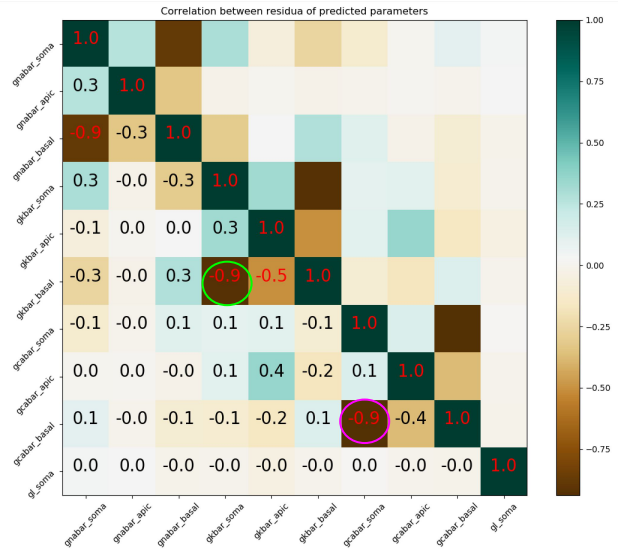


Model-693 predictions for a hard case

ML-predictions for 10-param **HH**







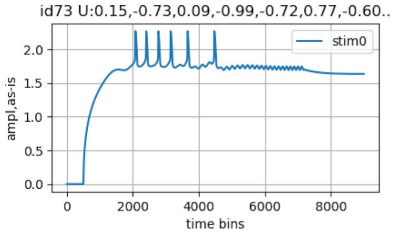
← Soma
Radius 10.5 um

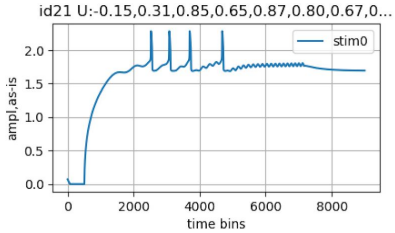
← 2 Basal dendrites of same properties
Length = 1/4th apical length
Radius = apical radius

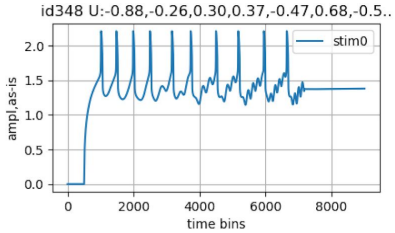
‘Adversarial’ predicting (setup=Mainen)

Input traces from Mainen 10param, scaled for ML, random sample

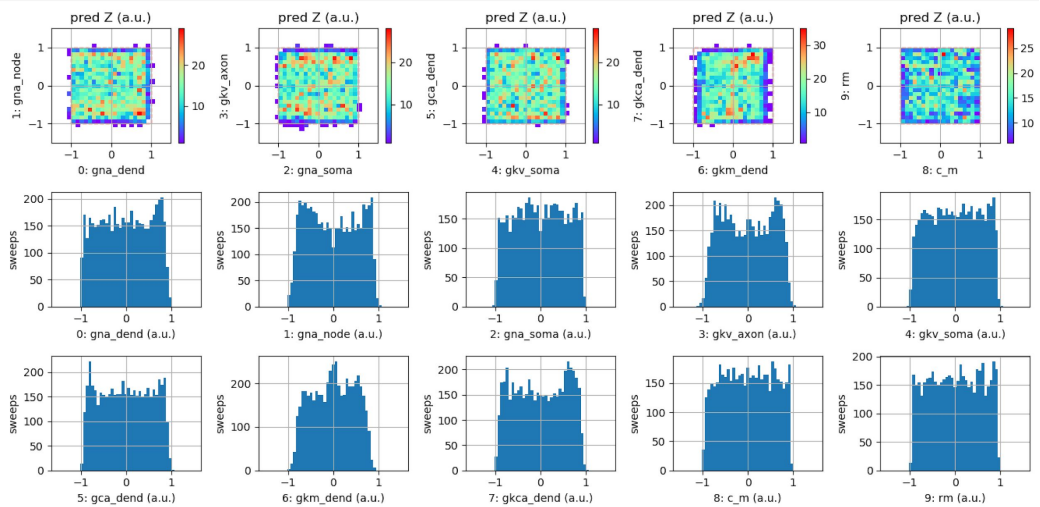
Model was trained of train+val data







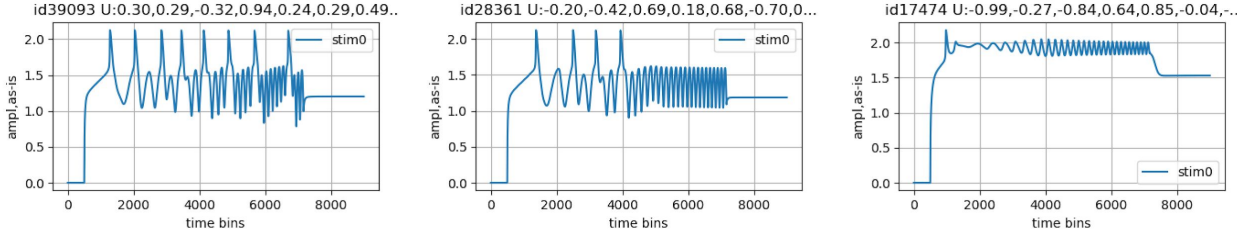
Predict 10 params for ‘test’ data.
No comparison to truth.
Show : 1D distribution of params and
2D correlations between pairs of params.
Uniform is good.



Note, the last layer of ML-model has the **activations 1.2* tanh**, yet predictions are confined to range [-1,1] - this is good too.

Adversarial predicting: input=Hodgkin–Huxley

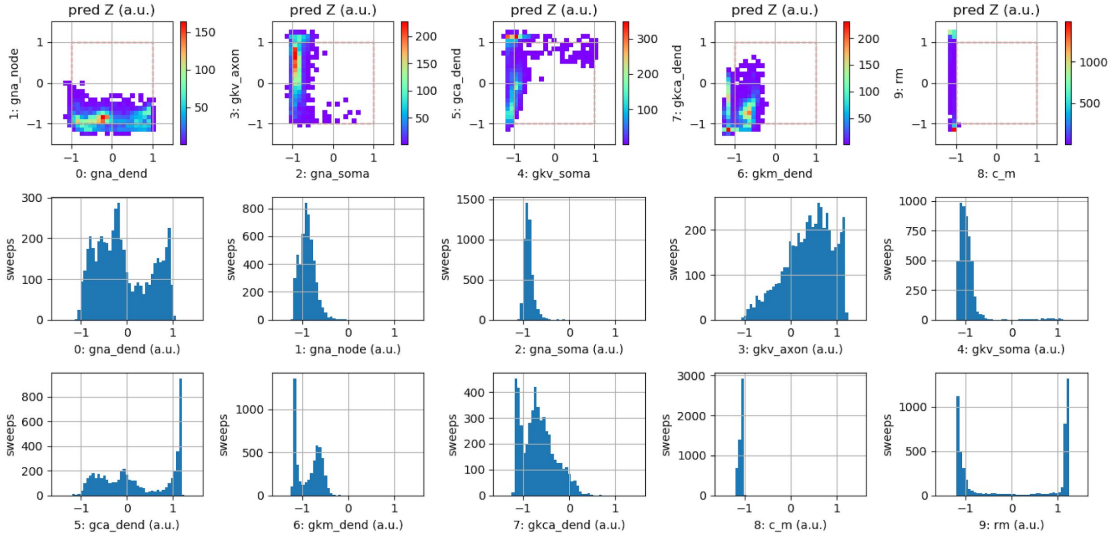
Input traces from HH_2dend_10param scaled for ML, random sample



Mainen-trained model

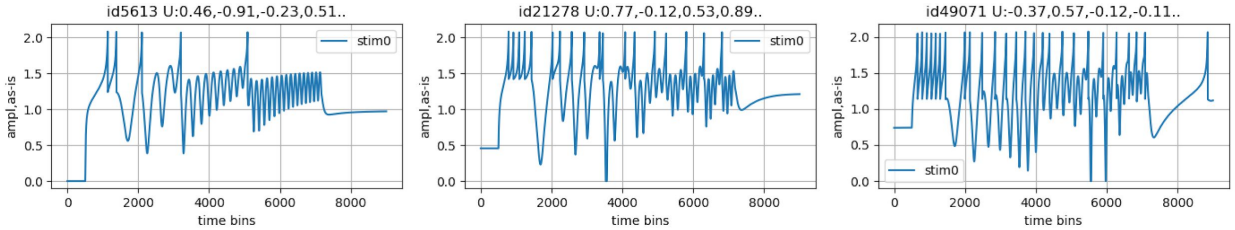
Predict 10 params for HH traces:

- Some predicted values are physical



Adversarial predicting: input=Izhikevich

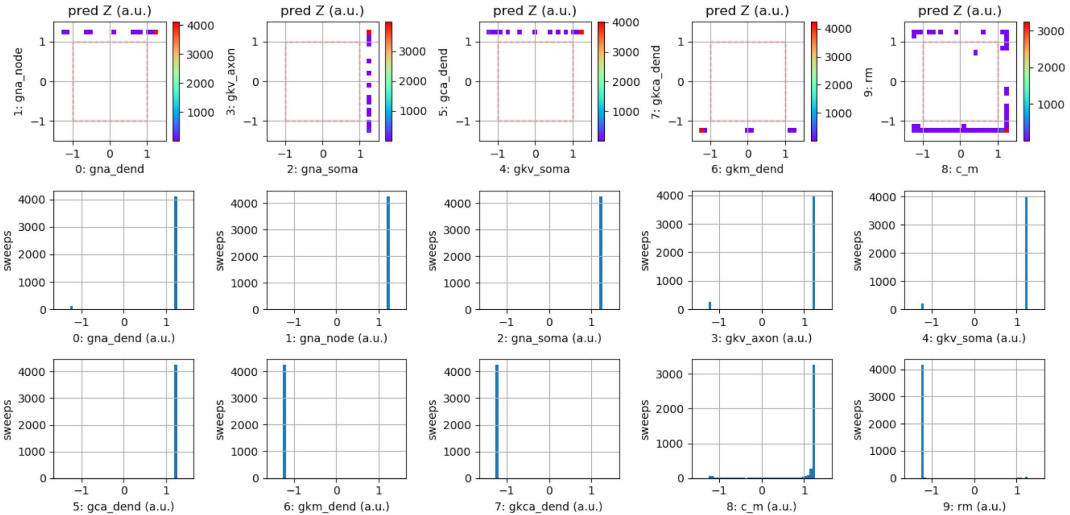
Input traces from Izhikevich 4 parameters , scaled for ML, random sample



Mainen-trained model

Predict 10 params for Izhikevich traces:

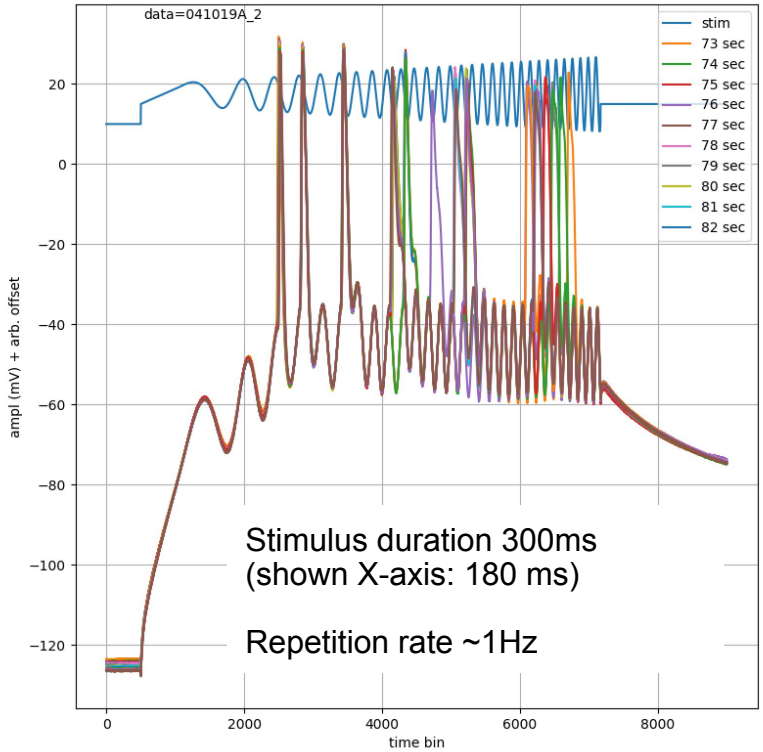
- Predicted values are mostly -1.2 or +1.2 - it is non-physical

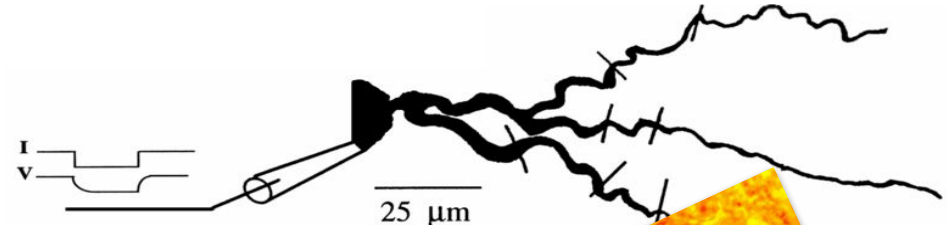


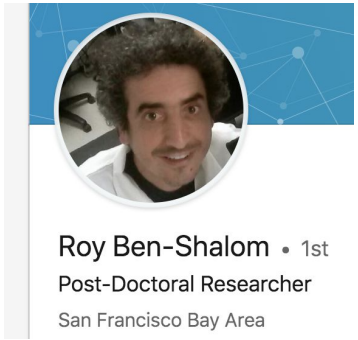
activations 1.2* tanh

Experimental data from pyramidal cell

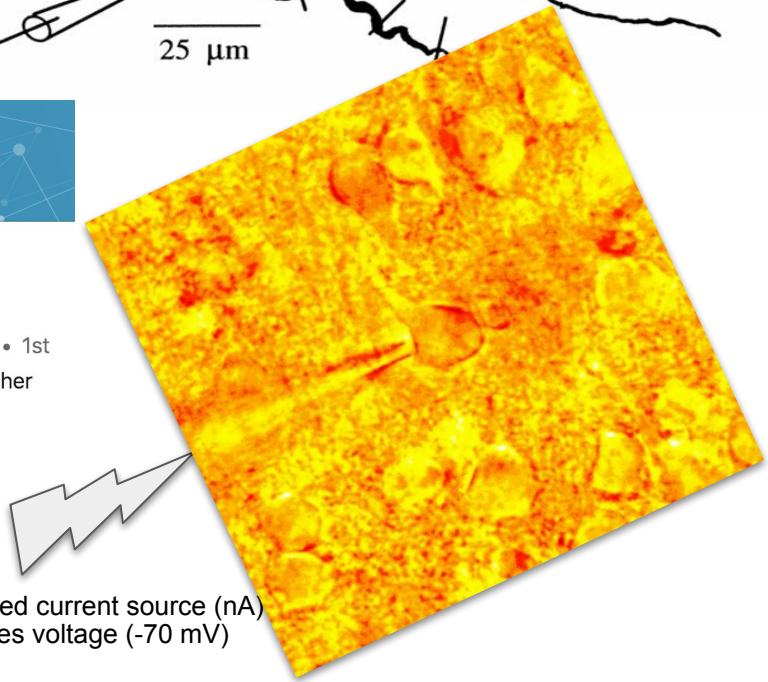
Current clamp recording from acute slices, find and measure from tufted pyramidal neuron







Roy Ben-Shalom • 1st
Post-Doctoral Researcher
San Francisco Bay Area



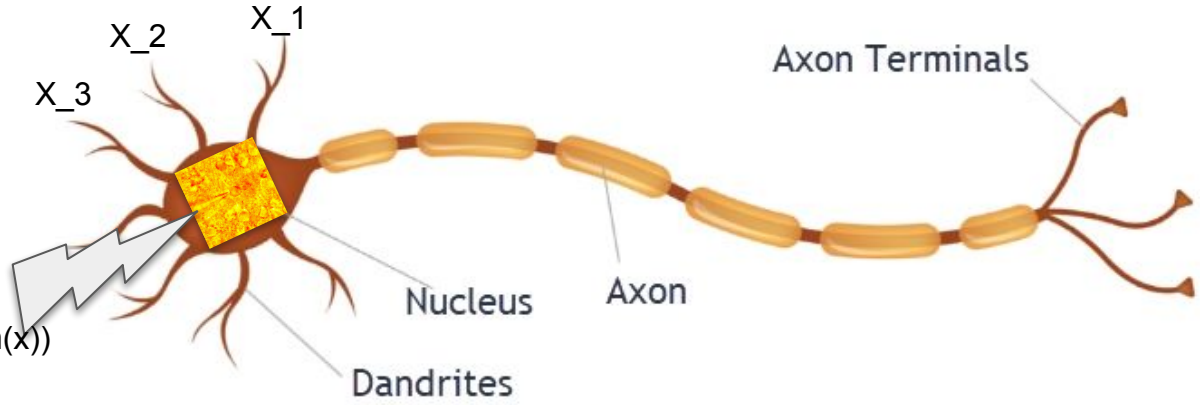
Current clamp:

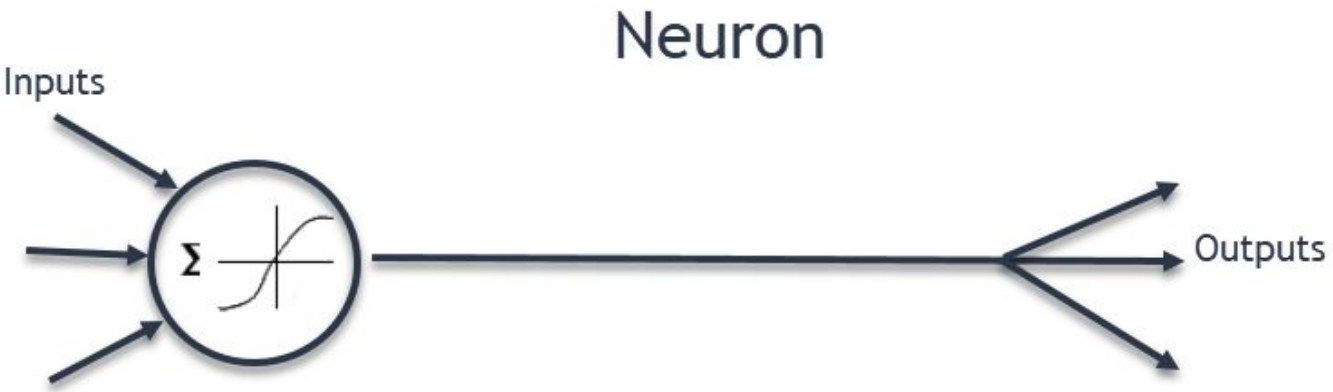
- INPUT Controlled current source (nA)
- Output Measures voltage (-70 mV)

Who is the observer?

Experiment
Current clamp:

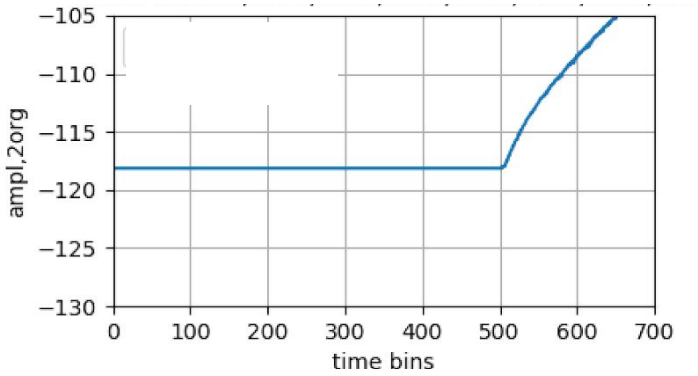
- INPUT X_5
- Output act(sum(x))

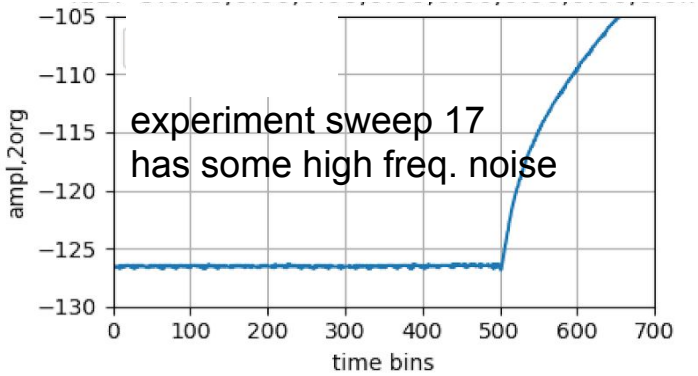




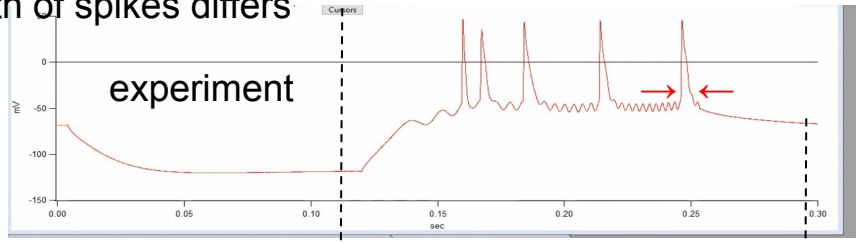
Experiment .NE. simulation

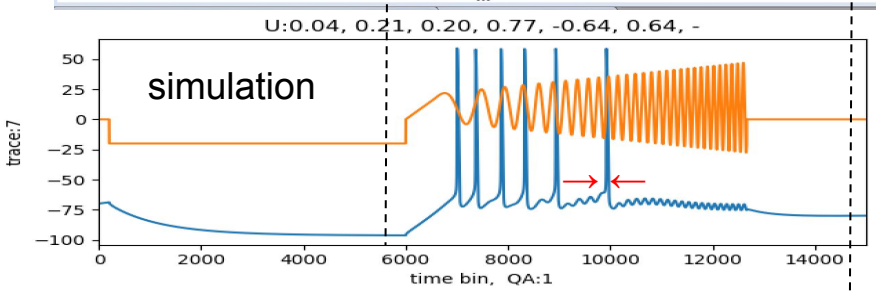
Zoom-in, experiment sweep 204

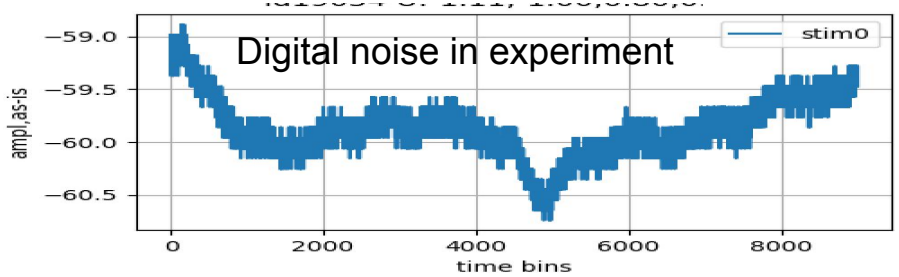




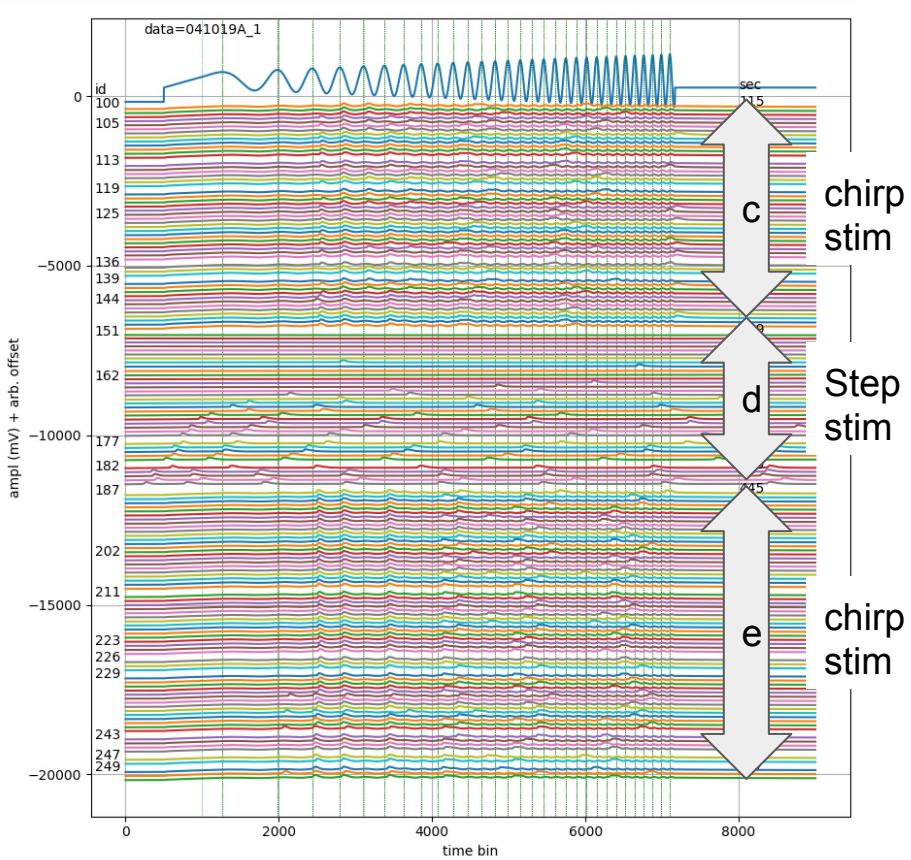
Width of spikes differs

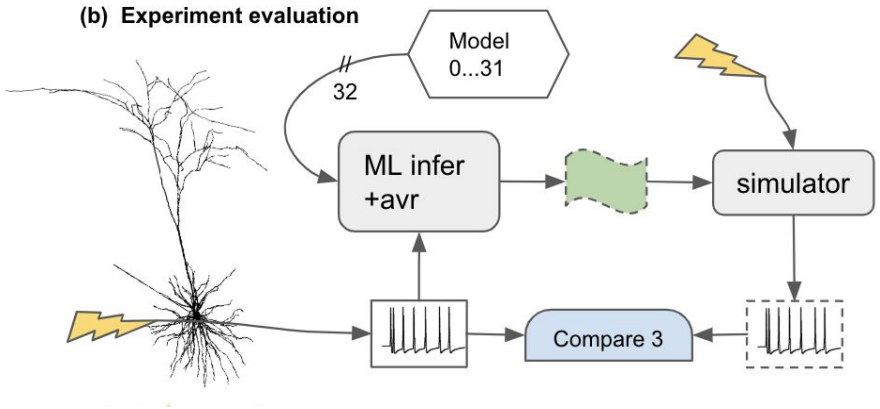






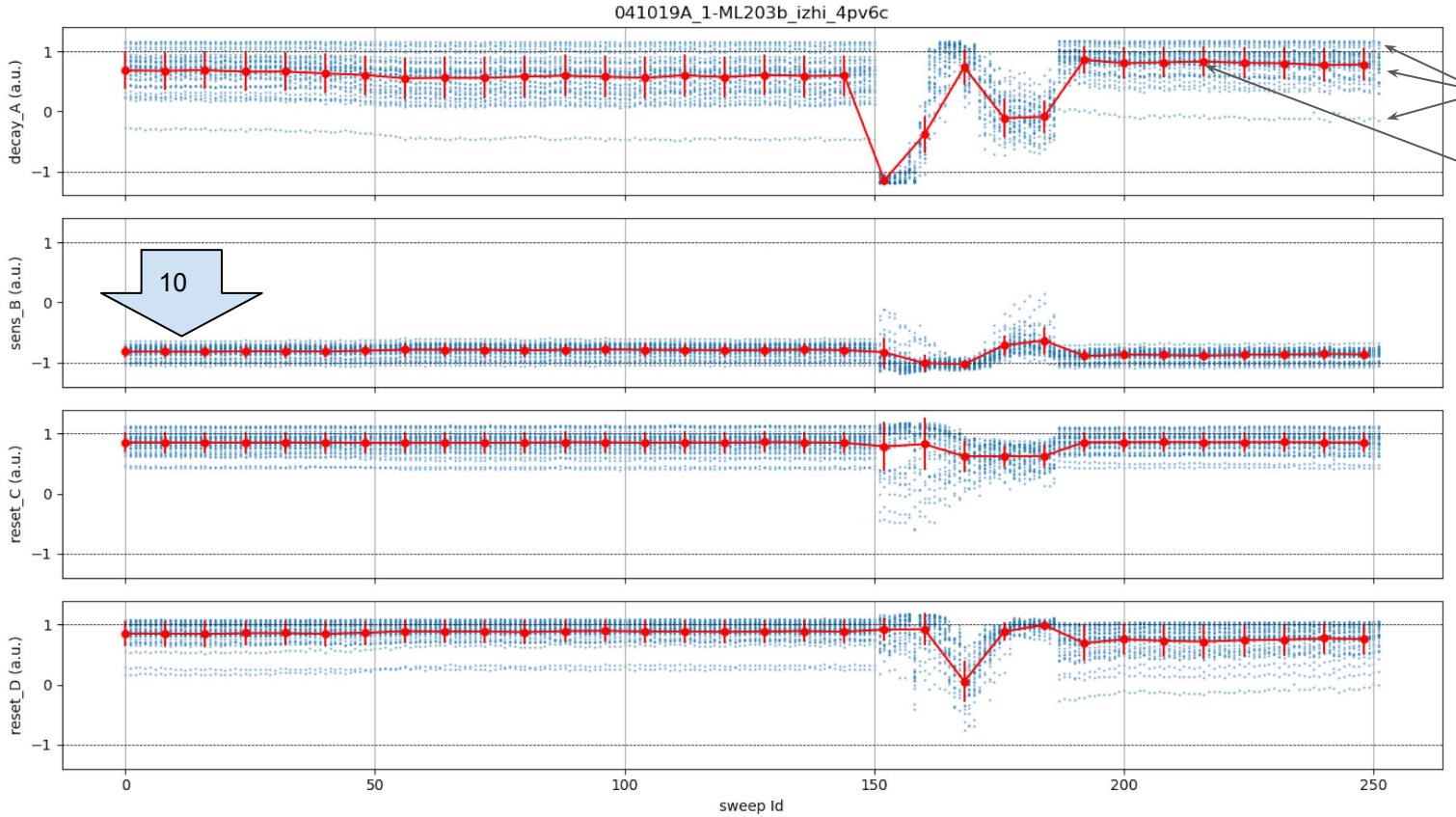
Experimental data analysis





Neuron (connectivity)
is changing as function of time !

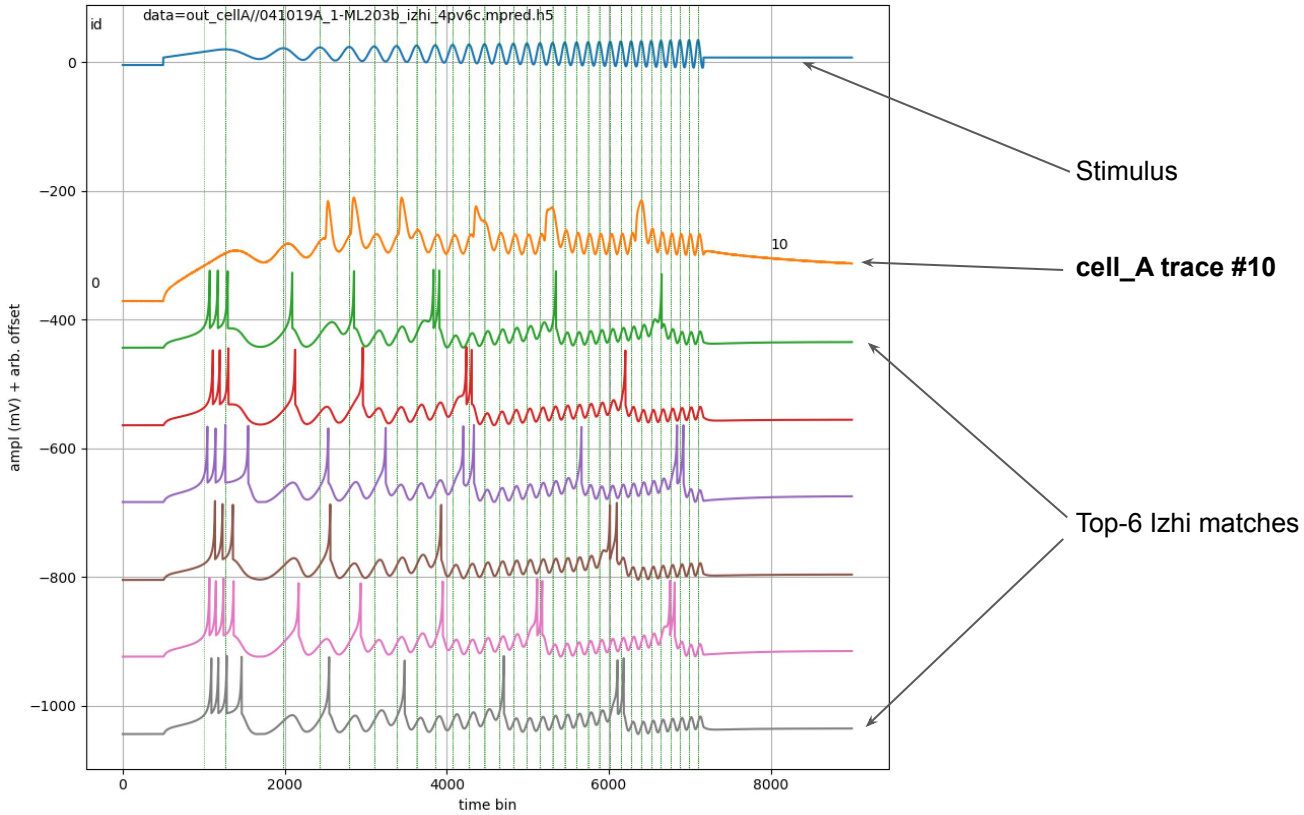
Predicting cell_A params using Izhi ML-model



32 ML predictions per sweep

Average +/- rms (plotted for every 8th sweep)

Izhi traces with for params <10% off ML pred.



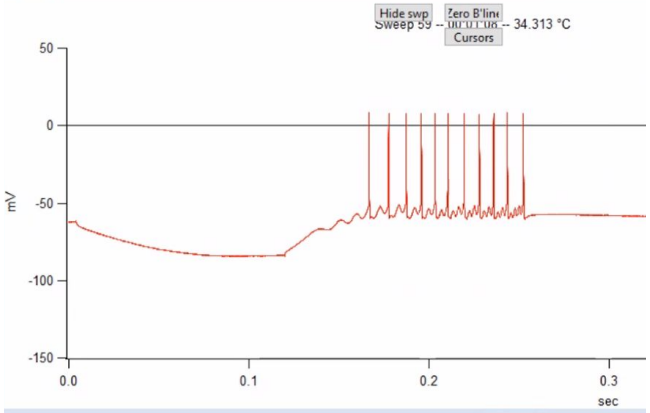
Stimulus

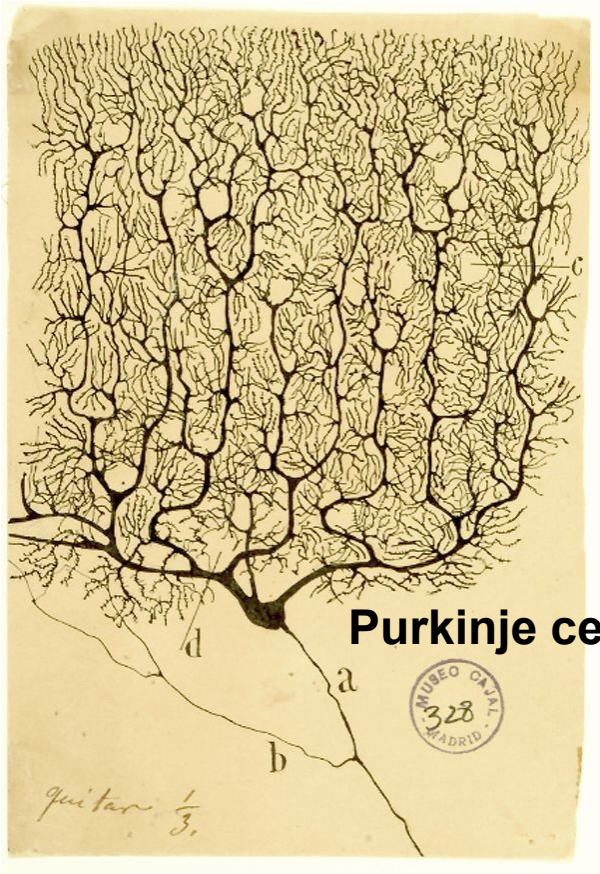
cell_A trace #10

Top-6 Izhi matches

Summary

- CNN-based ML-model is capable to regress time-dependent response into 10-param neuron model
- K-fold training provides a measure of error of prediction
- Improvement is needed:
  - Cross-model predictions
  - Tuning simulator to experimental data
  - More experimental data, more QA
  - Comparison of ML vs. standard methods (MOO, eFEL)





Purkinje cell

Backup

Accuracy vs. input size

*) the range [0.11, 0.29] sec, aka [5.5k,14.5k] , delta=9000bins = 180ms,

*) training data: mainein_10p26 data , chirp23a, use 1M traces

Method:

- Fix model: 203d
- Vary input size: 1M, 500K, 200k, 100k, 50k
- Train 32 models using 8 kfolds until convergence
- Compute test loss after the training for 50k events

Job ID	**Input size**	**Avr lossMSE**	Avr train time	Typical rmsErr gkv_axon
75914	**100k**	**0.056**	55 min	48%
76057	**200k**	**0.040**	90 min	38%
76016	**400k**	**0.030**	160 min	31%
75946	**800k**	**0.024**	280 min	27%

Example training on 800k traces