Workshop Series Summer 2023

CAMBAM

Centre for Applied Mathematics in Bioscience and Medicine

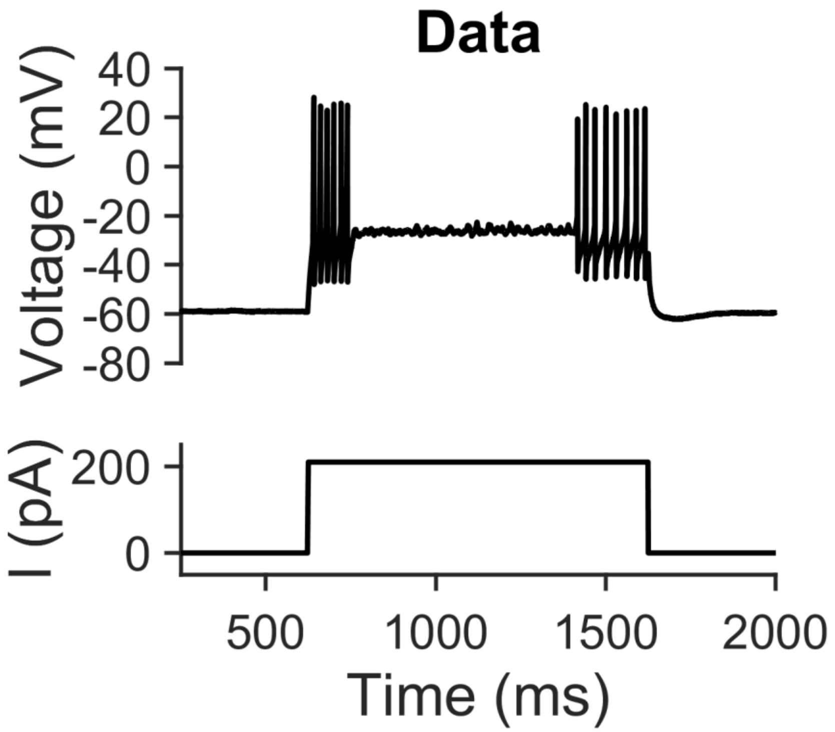
June 16th, 2023

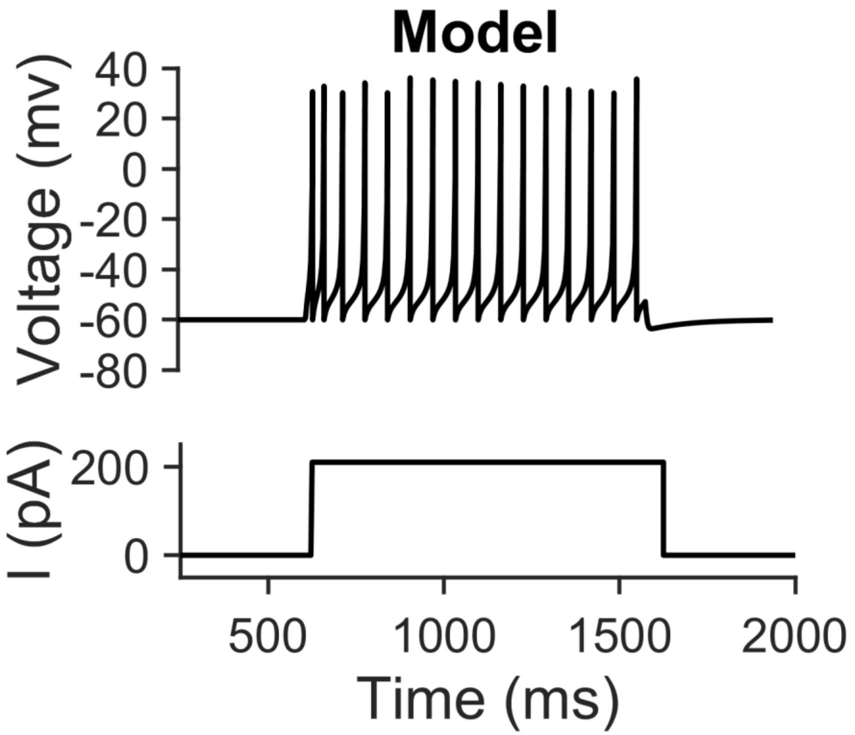
Exploring Single Neuron Excitability with Mathematical and Computational Models

By Niklas Brake and Nils Koch

Parameter optimization

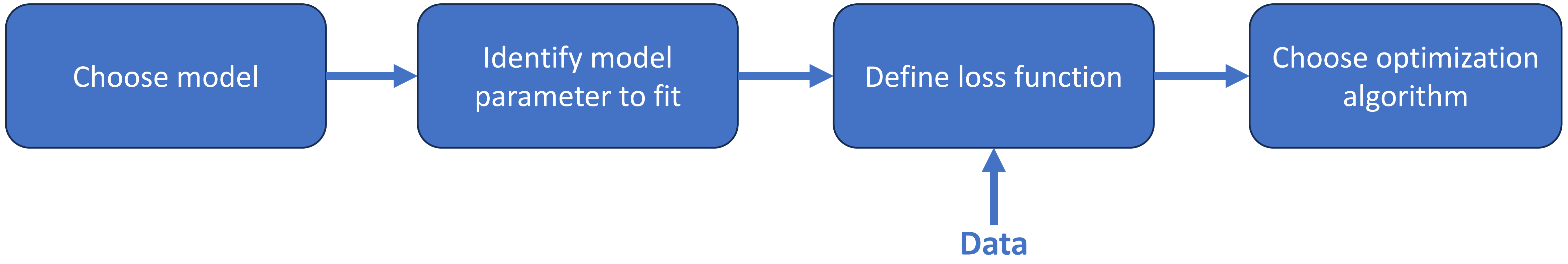
How to get the model to reproduce the data?





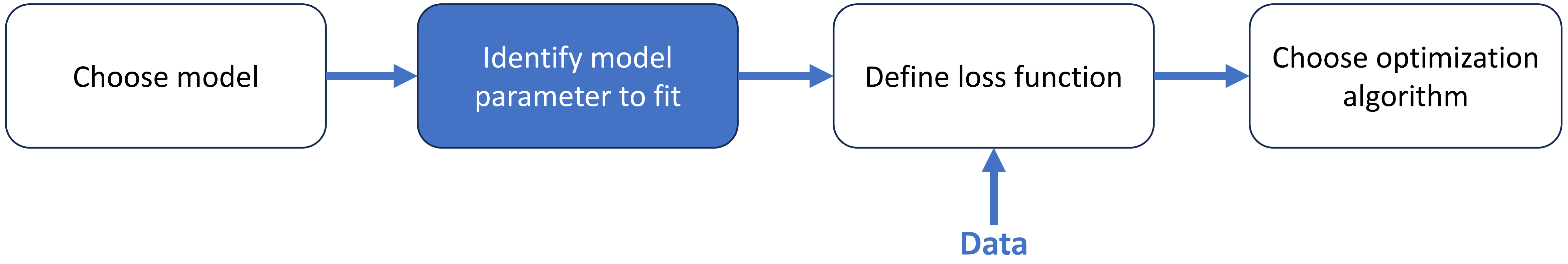
Parameter optimization

Process of reproducing experimental observations with a model


Choose model
Identify model parameter to fit
Define loss function
Choose optimization algorithm
Data

Parameter optimization

Process of reproducing experimental observations with a model


Choose model
Identify model parameter to fit
Define loss function
Choose optimization algorithm
Data

Parameter optimization: Identify parameters

Full Izhikevich model (better for fitting to data)

$$C\ \mathrm{d}v/\mathrm{dt} = k(v - v_r)(v - v_t) - u + I$$
$$\mathrm{d}u/\mathrm{dt} = a(b(v - v_r) - u)$$

$$\text{if } v \geq v_{peak}, \text{then}$$
$$v \leftarrow c, u \leftarrow u + d$$

I – injected current in picoamps (pA)
C – membrane capacitance in picofarads (pF)
k – scaling factor (nS)
v_r – resting membrane potential in millivolts (mV)
v_t – instantaneous threshold potential in millivolts (mV)
v_{peak} – peak amplitude triggering reset (mV)
a – timescale of recovery variable in 1/milliseconds (ms^{-1})
b – sensitivity of recovery variable (nS)
c – voltage after resetting (mV)
d – reset value of recovery variable (nS)

Some standard values:

$C = 100$ pF
$k = 1$ nS
$v_r = -60$ mV
$v_t = -40$ mV
$v_{peak} = 30$ mV

Parameter optimization: Identify parameters

Full Izhikevich model (better for fitting to data)

$$C\,\mathrm{d}v/\mathrm{dt} = k(v - v_r)(v - v_t) - u + I$$
$$\mathrm{d}u/\mathrm{dt} = a(b(v - v_r) - u)$$

$$\text{if } v \geq v_{peak}, \text{then}$$
$$v \leftarrow c, u \leftarrow u + d$$

Fixed parameters

I – Defined by experiment
C – 100 pF
k – 1 nS
v_r – -60 mV
v_t – -40 mV
v_{peak} – 30 mV

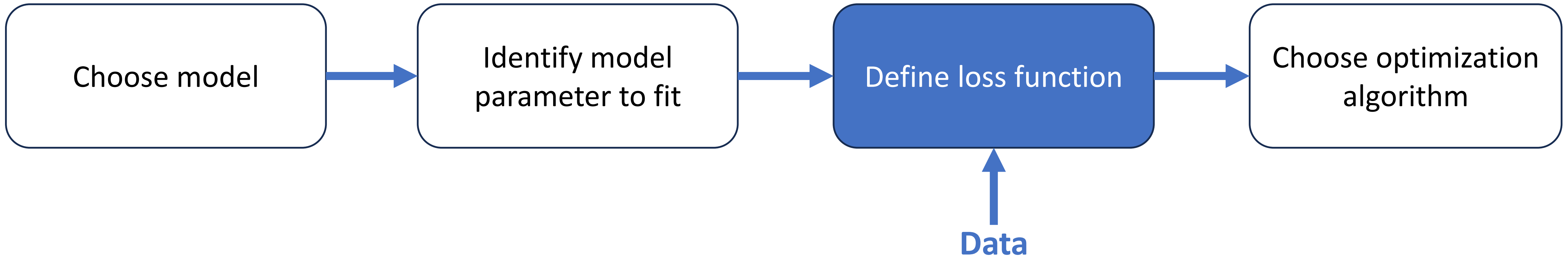
Parameters to fit

$a = ?,$	$a \in [0.001, 1]$
$b = ?,$	$b \in [-20, 20]$
$c = ?,$	$c \in [-80, -40]$
$d = ?,$	$d \in [0, 500]$

**Many algorithms require upper and lower bounds for the various parameters*

Parameter optimization

Process of reproducing experimental observations with a model


Choose model
Identify model parameter to fit
Define loss function
Choose optimization algorithm
Data

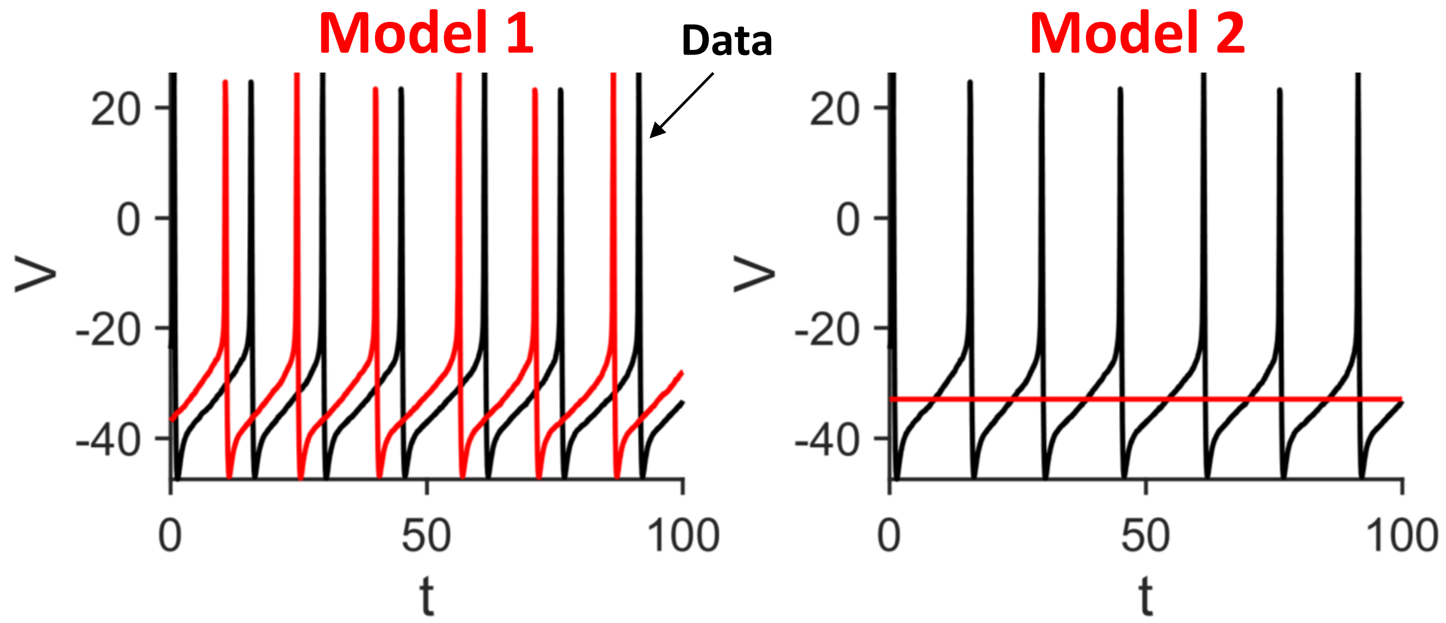
Parameter optimization: Loss functions

Example Loss Function #1

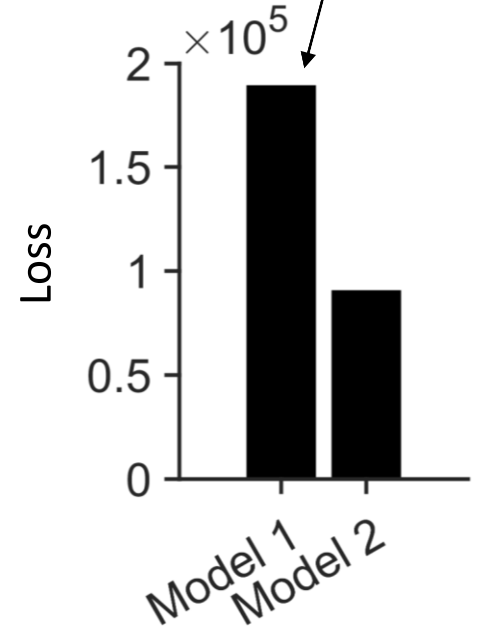
Most obvious choice for $\theta_i = V(t_i)$, giving us

$$L = \sum_{i} \left(V_{data}(t_i) - V_{model}(t_i)\right)^2$$

This function is good for many problem, but not for fitting neuronal firing data.



Loss function determines that Model 1 is worse



Parameter optimization: Loss functions

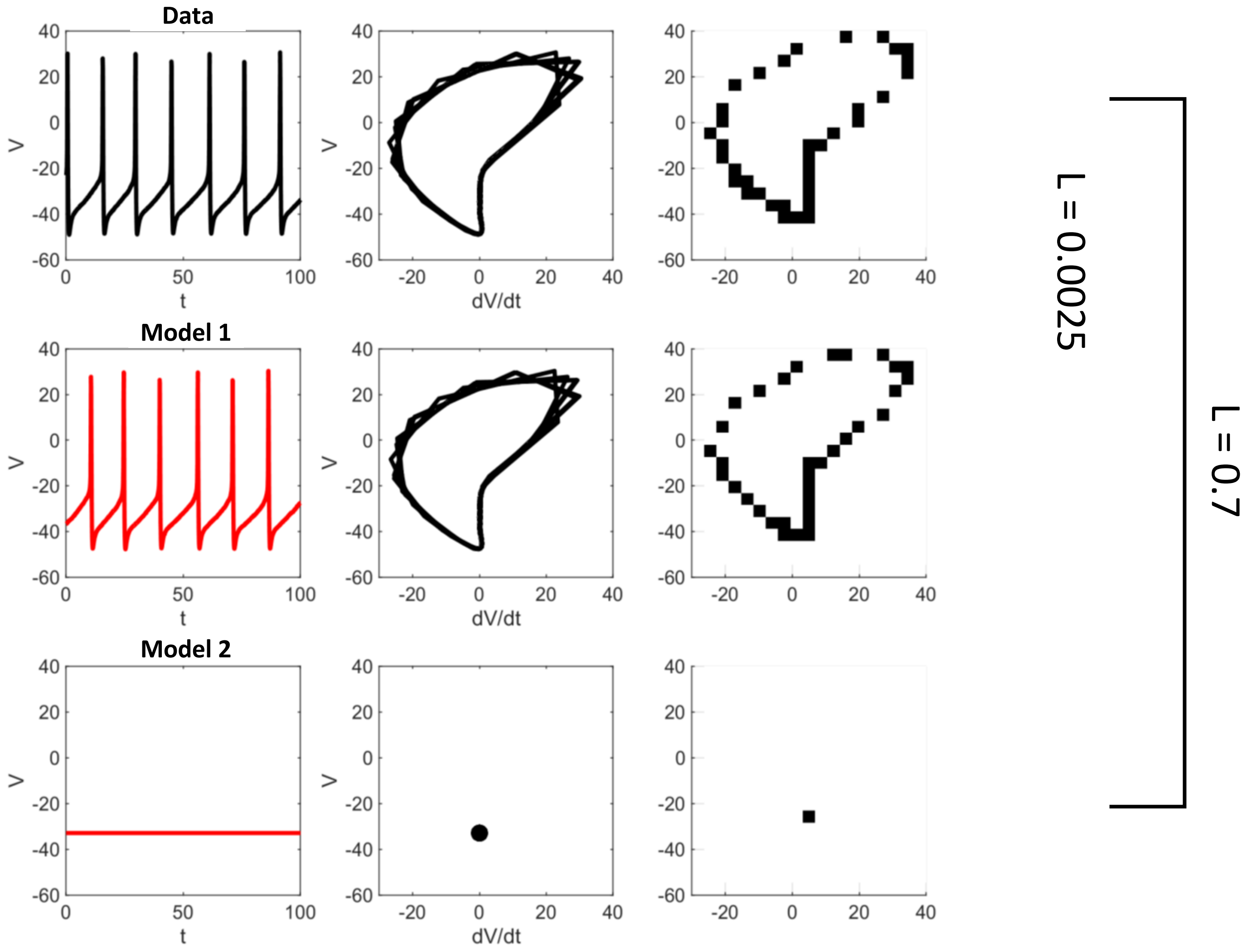
Example Loss Function #2

Phase plane fitting capture the shape of the action potential.

Plot V against dV/dt and compute density, i.e. a 2D histogram, on a $N_x \times N_y$ grid. Then define the loss function as

$$L = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} \left(\frac{data_{ij}}{N_{data}} - \frac{model_{ij}}{N_{model}} \right)^2$$

This allows fitting the precise shape of the action potential and is particularly relevant for Hodgkin-Huxley modelling.



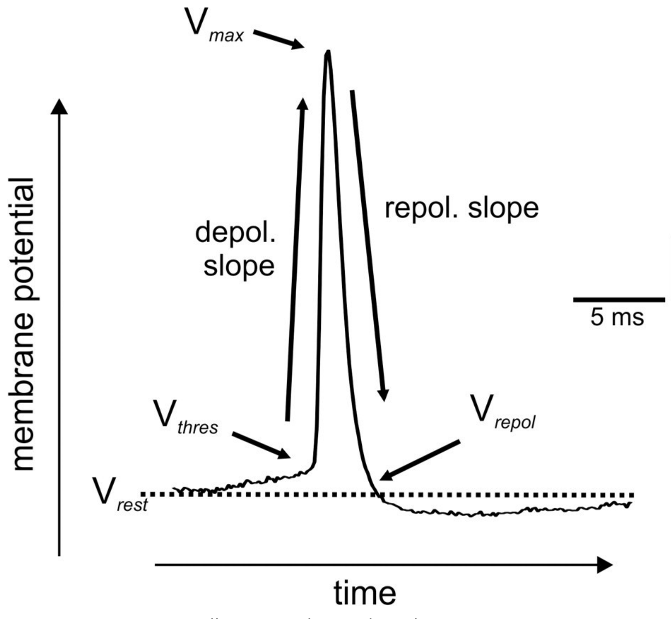
Parameter optimization: Loss functions

Example Loss Function #3

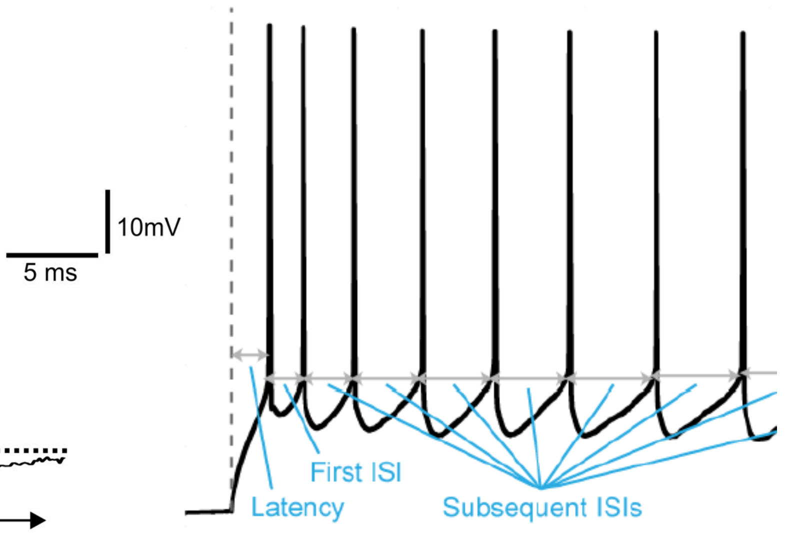
Extract firing features from data and compute loss function on these.

$$\theta_1 = V_{thres}$$
$$\theta_2 = V_{rest}$$
$$\theta_3 = \text{Latency to first spike}$$
$$\ldots$$

$$L = \sum_i \left(\theta_i - \hat{\theta}_i\right)^2$$



Allen Brain Electrophysiology Overview



Trombin, Federica & Gnatkovsky, Vadym & de Curtis, Marco. (2011). Changes in action potential features during focal seizure discharges in the entorhinal cortex of the in vitro isolated guinea pig brain. Journal of neurophysiology. 106. 1411-23.

A useful modification:

Performance tends to be better with the following modified loss function

Different weights for each feature

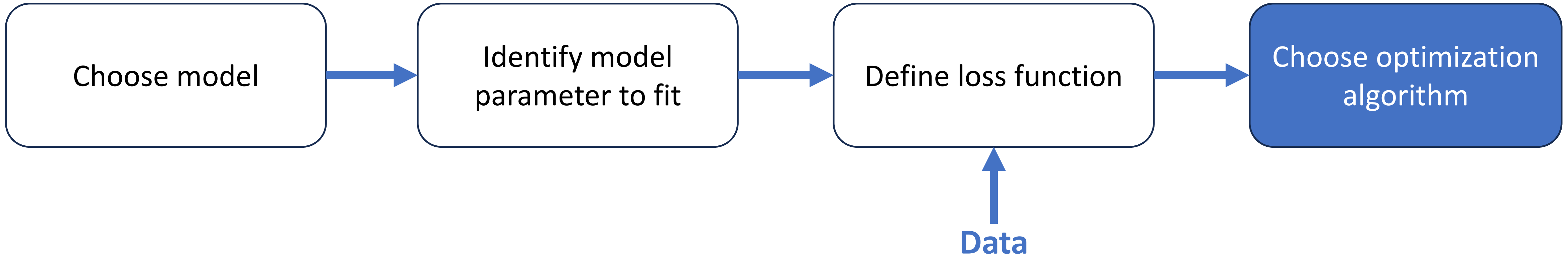
$$L = \sum_i w_i \frac{\left(\theta_i - \hat{\theta}_i\right)^2}{\sigma_i}$$

Standard deviation of experimentally measured featured

One must still determine the weights w_i as well as *which* features to fit. These decisions are often taken ad-hoc and do not have a single answer.

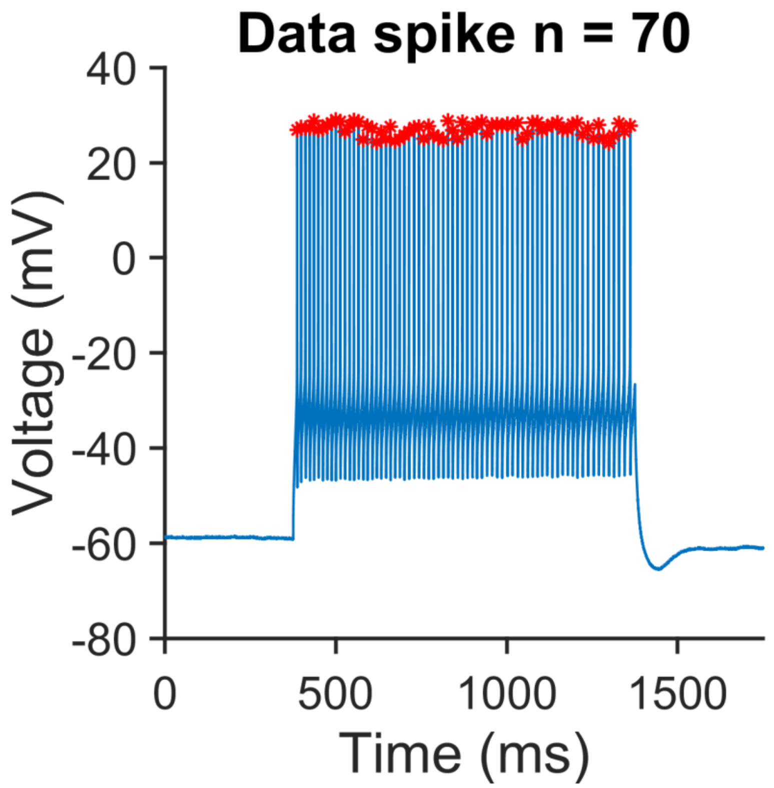
Parameter optimization

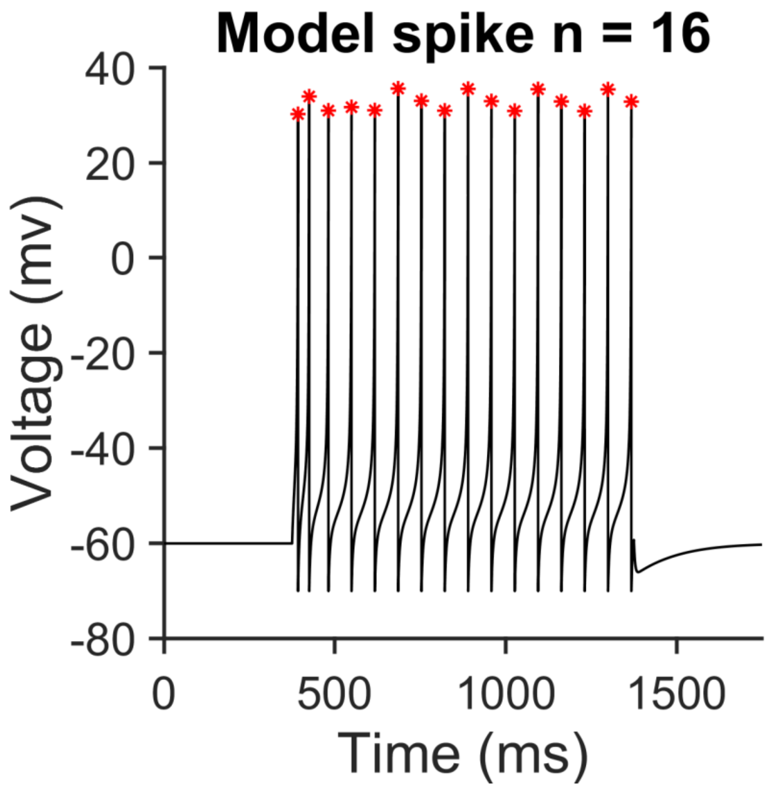
Process of reproducing experimental observations with a model


Choose model
Identify model parameter to fit
Define loss function
Choose optimization algorithm
Data

Parameter optimization: Searching parameter space

Suppose we want an Izhikevich model to match the total number of action potentials fired in response to a current injection

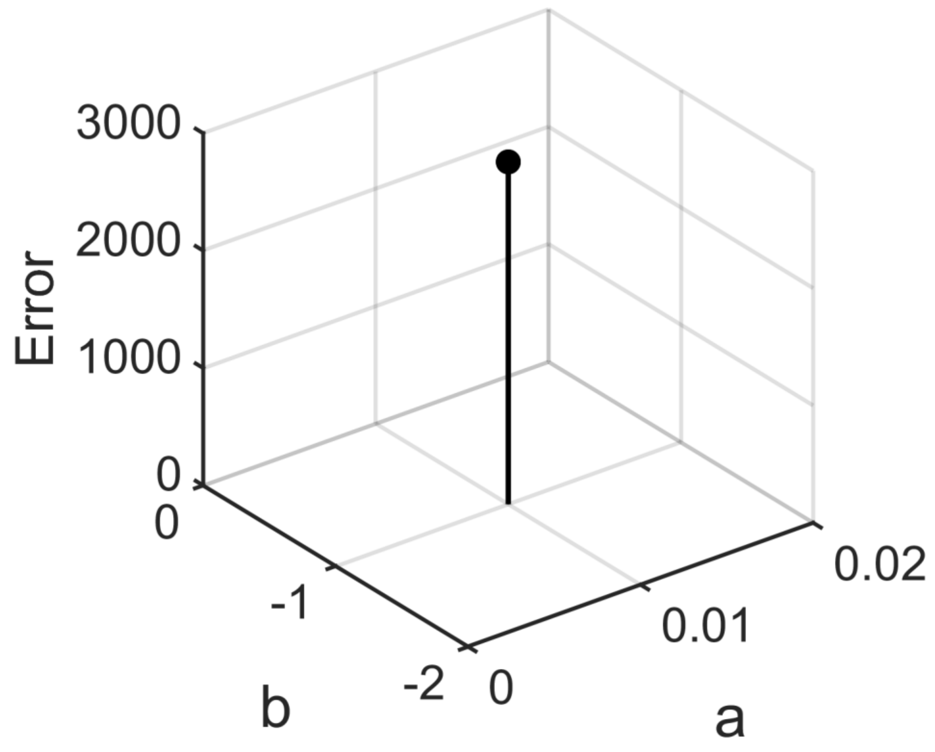




Loss$(a = 0.01, b = -1, c = -60, d = 70) \rightarrow (70-16)^2 = 2916$

Parameter optimization: Searching parameter space

Loss(a = 0.01, b = -1, c = -60, d = 70) → $(70-16)^2 = 2916$



We now know what the error is at a single point in *parameter space*.

Given our bounds from a previous slide, our job is to search a 4D hypercube with the following ranges:

$$a \in [0.001,1],\ b \in [-20,20],\ c \in [-80,-40],\ \ d \in [0,500]$$

and find the point (a,b,c,d) that minimizes the error. There are many strategies, or algorithms, to do this.

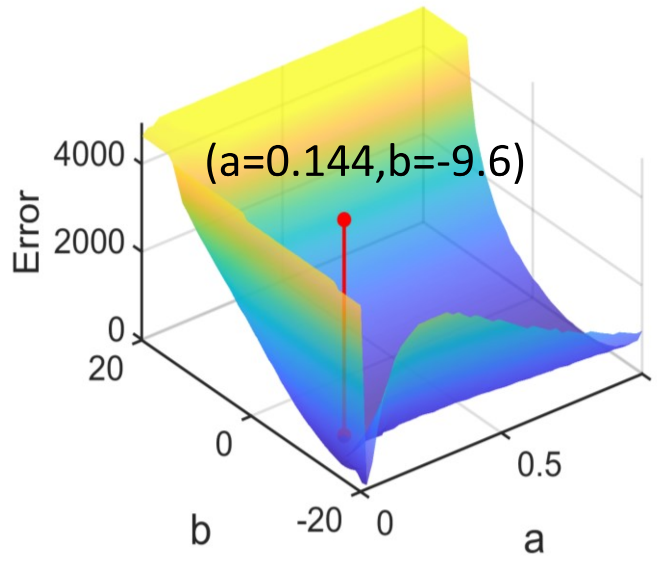
Parameter optimization: Searching parameter space

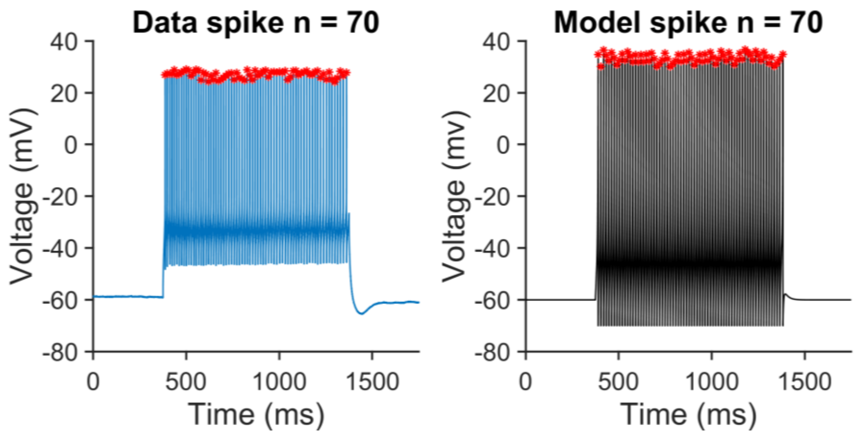
Example Algorithm #1: Grid Search

Brute force approach. Let's split our entire parameter space into equally spaced grids and compute the model at every point.

To the right, I sampled $a \in [0.001,1]$ and $b \in [-20,20]$ from a 50x50 array of values (keeping $c = 70$ and $d = 100$ fixed). This took ~5 s and we found an optimal solution.

- To explore the full 4D space, i.e., a 50x50x50x50 array of values, this would have taken 3.5 hours. Also, to fit the full F-I curve, we would need to repeat this for ~10 current steps, which would take ~35 hours.
- If we want to fit more complex features, such as AP-width, steady-state voltage, adaptation, etc., this approach becomes quickly impractical.
- However, the example to the right shows that in certain cases it is an effective method.





Parameter optimization: Searching parameter space

Example Algorithm #2: Gradient Descent

From the surface plot to the right, one can see that the "fitness landscape" isn't completely random but has some smoothness to it.

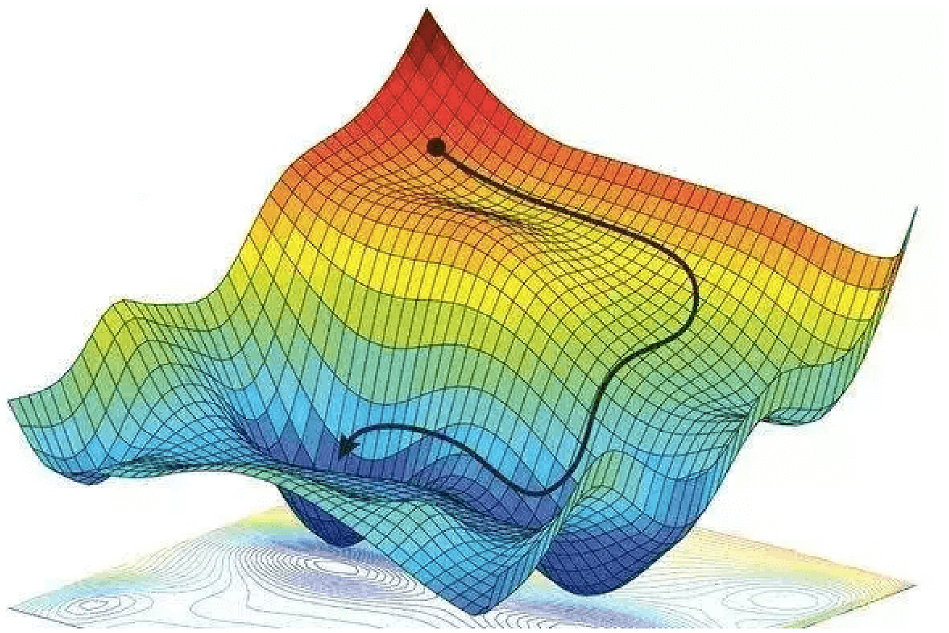
Gradient descent exploits this smoothness by estimating the derivative of the error function at each point and moves in the direction of fastest descent.

Pros

- No longer need to waste time computing model for parameters that are likely to produce bad fits

Cons

- Choosing an appropriate step size is critical for good algorithm performance
- Algorithm converges to the minimum of the function closest to the starting point. This is not necessarily the best solution in the entire parameter space.

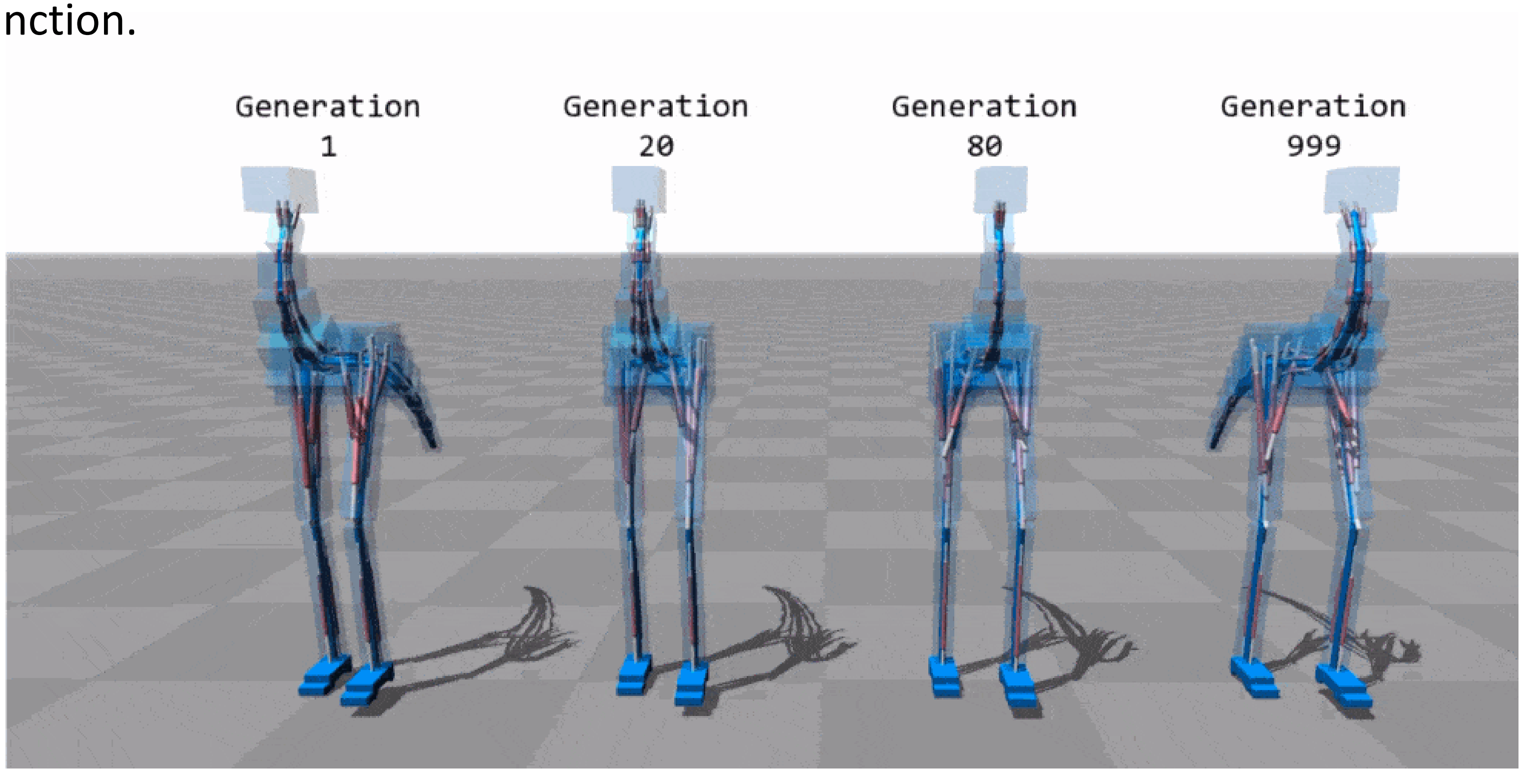


easyai.tech/en/ai-definition/gradient-descent/

Parameter optimization: Searching parameter space

Example Algorithm #3: Evolutionary (Genetic) Algorithm

Generate a population of models and let evolution do its thing! The models will response to “environmental pressure” which here is a penalty to model survival for the errors defined by the loss function.



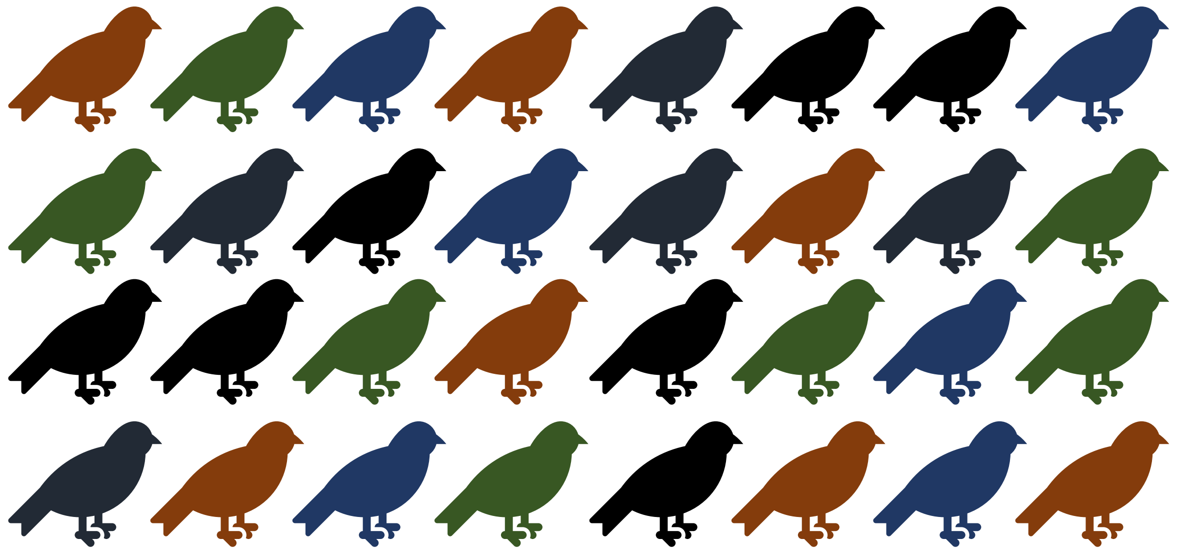
Animation source: “Flexible Muscle-Based Locomotion for Bipedal Creatures” — Thomas Geijtenbeek

Parameter optimization: Searching parameter space

Example Algorithm #3: Evolutionary (Genetic) Algorithm

Step 1: Initial population of models. For example, bird i represents the parameter set (a_i, b_i, c_i, d_i)**.**

Generation #1

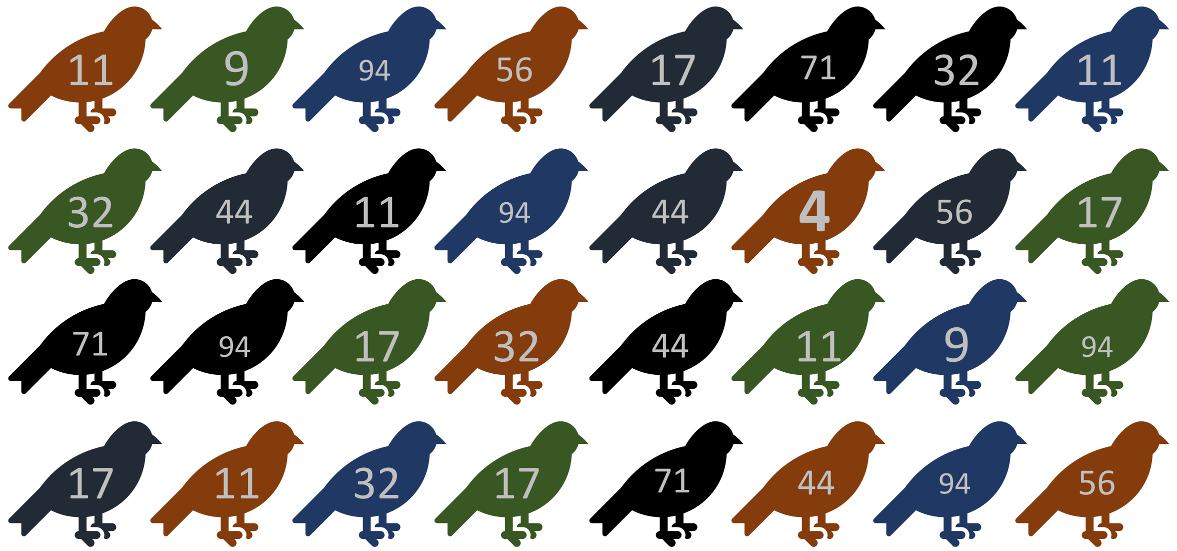


Parameter optimization: Searching parameter space

Example Algorithm #3: Evolutionary (Genetic) Algorithm

Step 2: Simulate models and evaluate each "individual" with loss function.

Generation #1

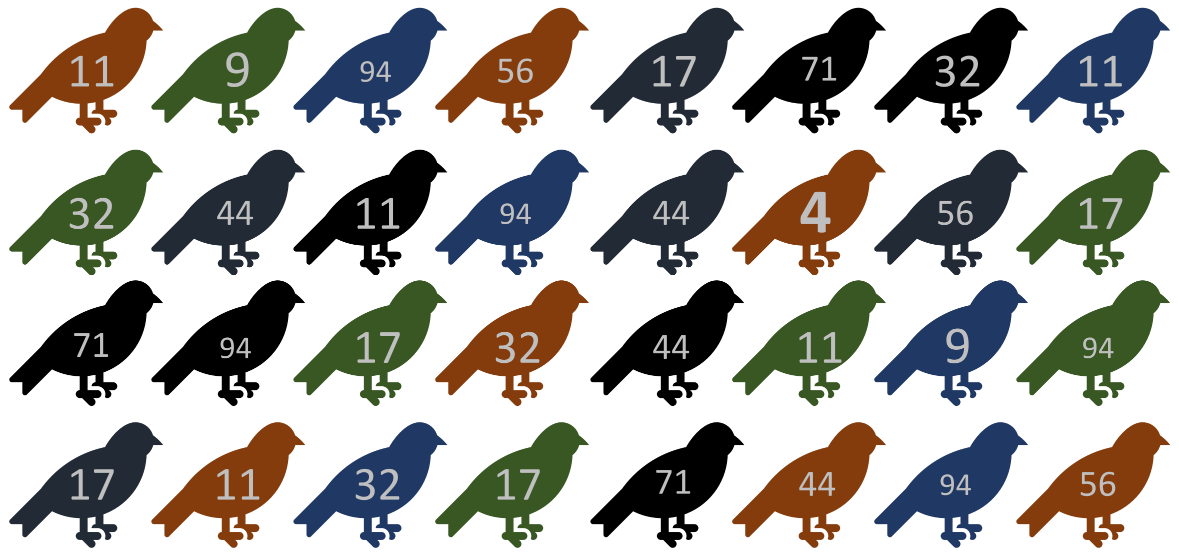


Parameter optimization: Searching parameter space

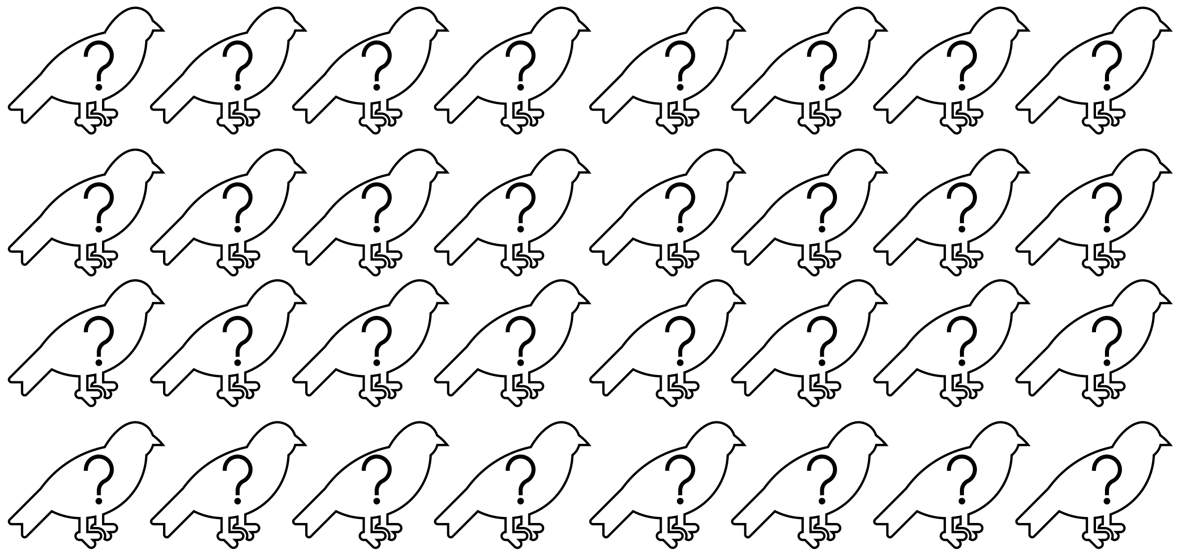
Example Algorithm #3: Evolutionary (Genetic) Algorithm

Step 3: Move to the next generation. To do this, we will generate children from the current generation of models.

Generation #1



Generation #2

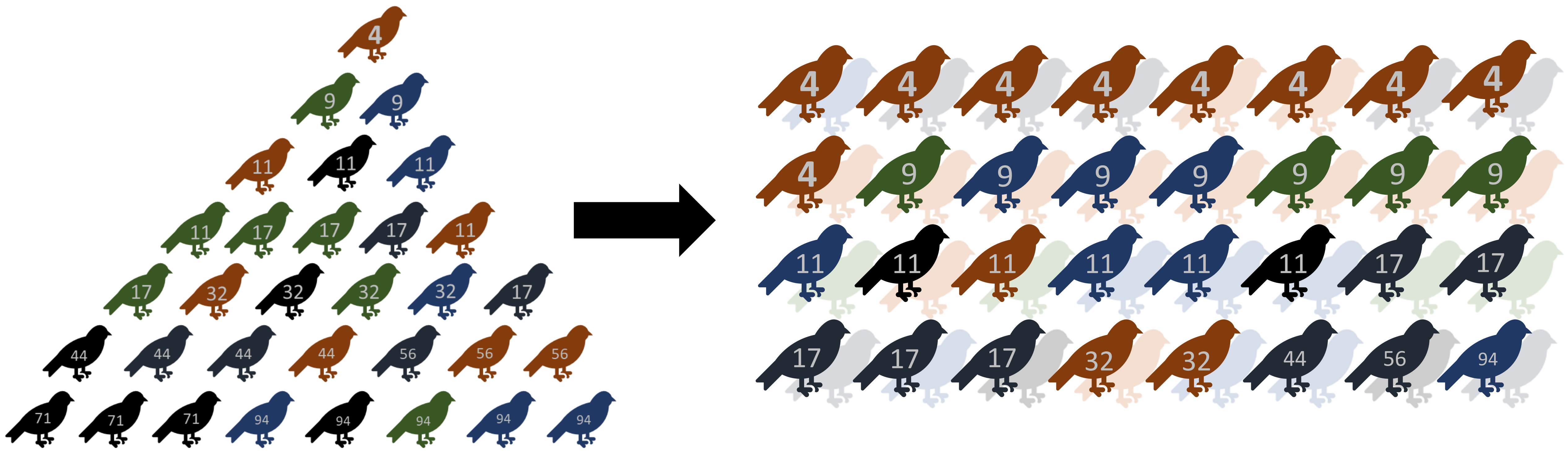


Parameter optimization: Searching parameter space

<u>Example Algorithm #3: Evolutionary (Genetic) Algorithm</u>

Step 3: Move to the next generation. To do this, we will generate children from the current generation of models.

- Choose individuals to be parents depending on their fitness. An individual can be a parent more than once.

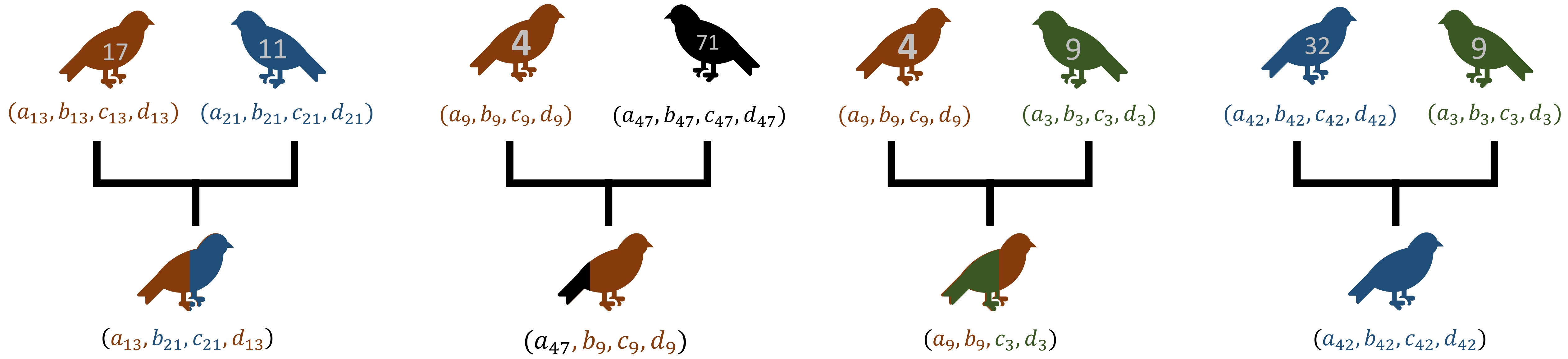


Parameter optimization: Searching parameter space

Example Algorithm #3: Evolutionary (Genetic) Algorithm

Step 3: Move to the next generation. To do this, we will generate children from the current generation of models.

- Choose individuals to be parents depending on their fitness. An individual can be a parent more than once.
- Randomly mix the parameters of parents and add mutations

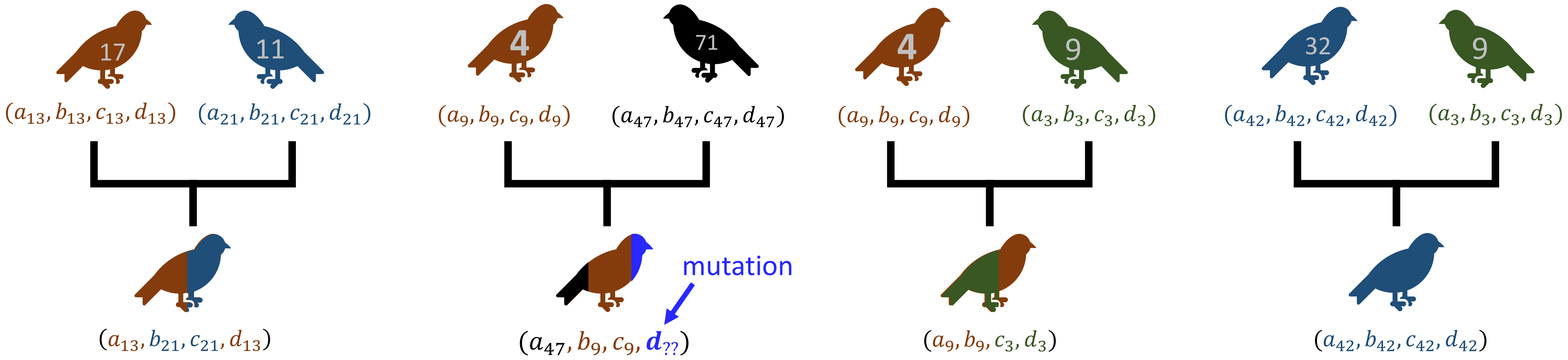

17
11
4
71
4
9
32
9
$(a_{13}, b_{13}, c_{13}, d_{13})$
$(a_{21}, b_{21}, c_{21}, d_{21})$
(a_9, b_9, c_9, d_9)
$(a_{47}, b_{47}, c_{47}, d_{47})$
(a_9, b_9, c_9, d_9)
(a_3, b_3, c_3, d_3)
$(a_{42}, b_{42}, c_{42}, d_{42})$
(a_3, b_3, c_3, d_3)
$(a_{13}, b_{21}, c_{21}, d_{13})$
(a_{47}, b_9, c_9, d_9)
(a_9, b_9, c_3, d_3)
$(a_{42}, b_{42}, c_{42}, d_{42})$

Parameter optimization: Searching parameter space

Example Algorithm #3: Evolutionary (Genetic) Algorithm

Step 3: Move to the next generation. To do this, we will generate children from the current generation of models.

- Choose individuals to be parents depending on their fitness. An individual can be a parent more than once.
- Randomly mix the parameters of parents and add mutations



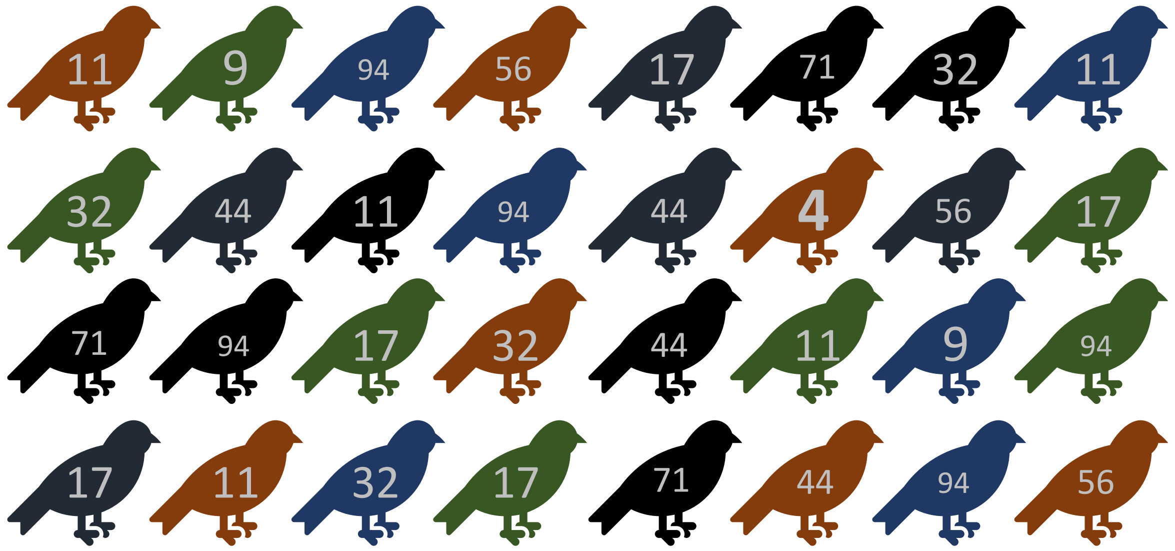
17
11
4
71
4
9
32
9
$(a_{13}, b_{13}, c_{13}, d_{13})$
$(a_{21}, b_{21}, c_{21}, d_{21})$
(a_9, b_9, c_9, d_9)
$(a_{47}, b_{47}, c_{47}, d_{47})$
(a_9, b_9, c_9, d_9)
(a_3, b_3, c_3, d_3)
$(a_{42}, b_{42}, c_{42}, d_{42})$
(a_3, b_3, c_3, d_3)
mutation
$(a_{13}, b_{21}, c_{21}, d_{13})$
$(a_{47}, b_9, c_9, \boldsymbol{d}_{??})$
(a_9, b_9, c_3, d_3)
$(a_{42}, b_{42}, c_{42}, d_{42})$

Parameter optimization: Searching parameter space

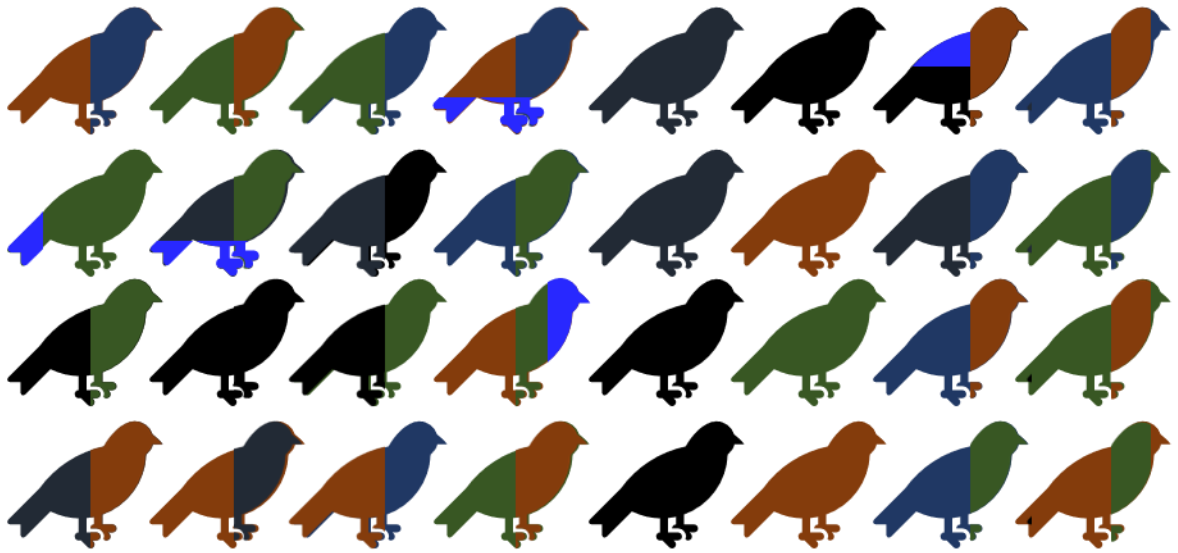
Example Algorithm #3: Evolutionary (Genetic) Algorithm

Step 3: Move to the next generation. To do this, we will generate children from the current generation of models.

Generation #1



Generation #2

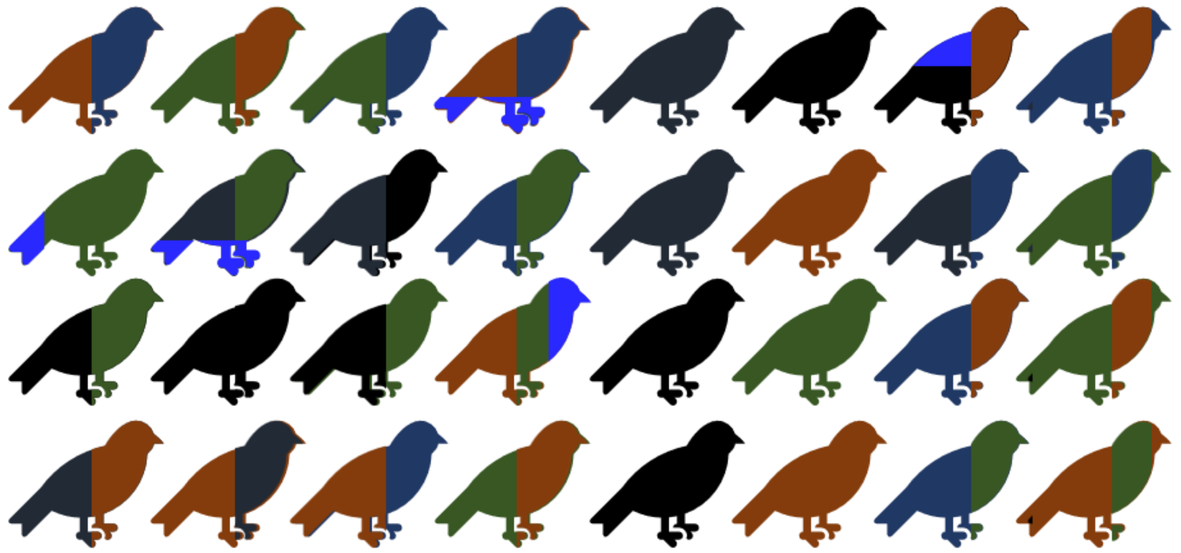


Parameter optimization: Searching parameter space

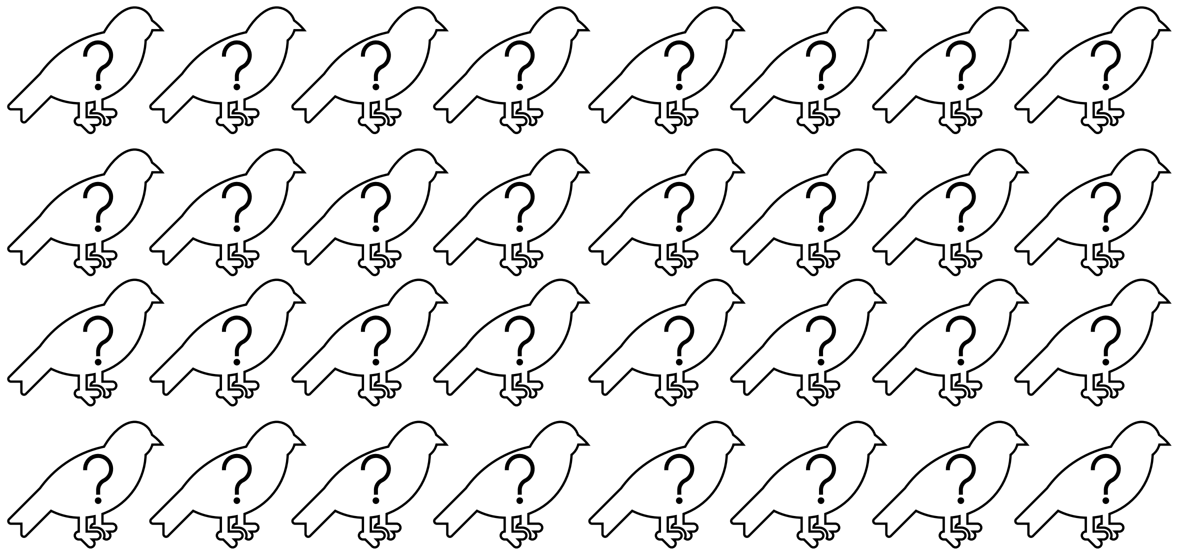
Example Algorithm #3: Evolutionary (Genetic) Algorithm

Step 4: Repeat until model stops improving (or rate of improvement slows below a certain level)

Generation #2



Generation #3



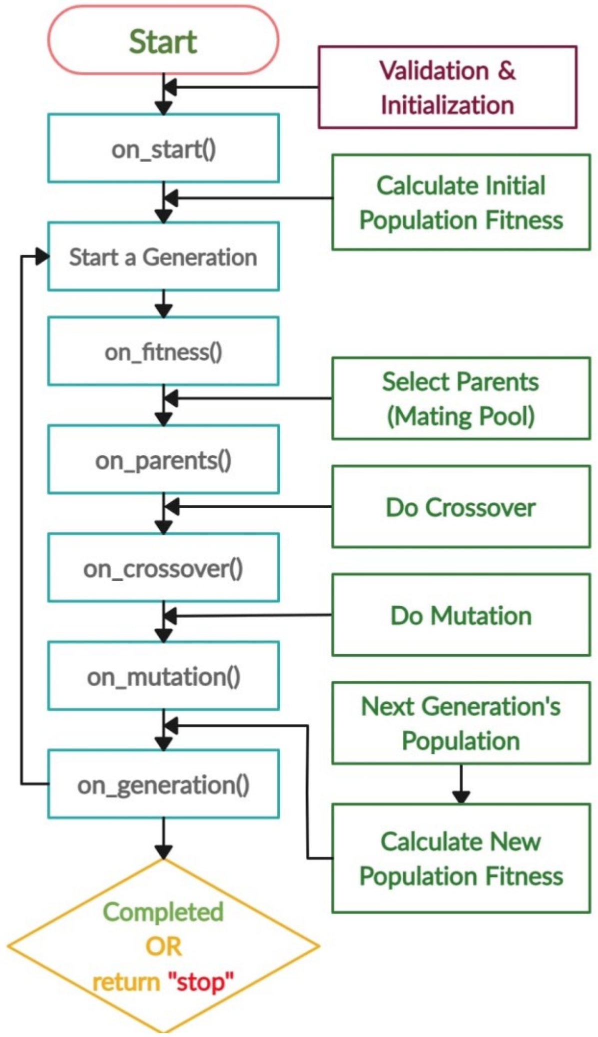
Parameter optimization: Searching parameter space

Example Algorithm #3: Evolutionary (Genetic) Algorithm

These sorts of algorithm tend to have a lot of different variants with a lot of parameters. However, they all follow the same evolution framework.

Examples of algorithm variations:

- **Elitism:** The best individual always continues to the next generation unaltered (*beneficial for stability so that a good model doesn't get lost to the randomness of evolution*)
- **Parent selection:** Parents can be chosen with various algorithms, based on fitness ranking, absolute fitness value, "tournament selection." E.g., all the parents come from the top 10 individual from the previous generation.
- **Gene mixing**: Genes can be mixed in various ways. E.g. genes can undergo "crossing over" where the first k parameters are taken from parent 1 and $m - k$ parameters are taken from parent 2 to select all m parameters.
- **Mutations**: Mutations can be introduced by random noise added to all parameters, some of the parameters, etc.


Start
Validation & Initialization
on_start()
Calculate Initial Population Fitness
Start a Generation
on_fitness()
Select Parents (Mating Pool)
on_parents()
Do Crossover
on_crossover()
Do Mutation
on_mutation()
Next Generation's Population
on_generation()
Calculate New Population Fitness
Completed OR return "stop"

Parameter optimization: Searching parameter space

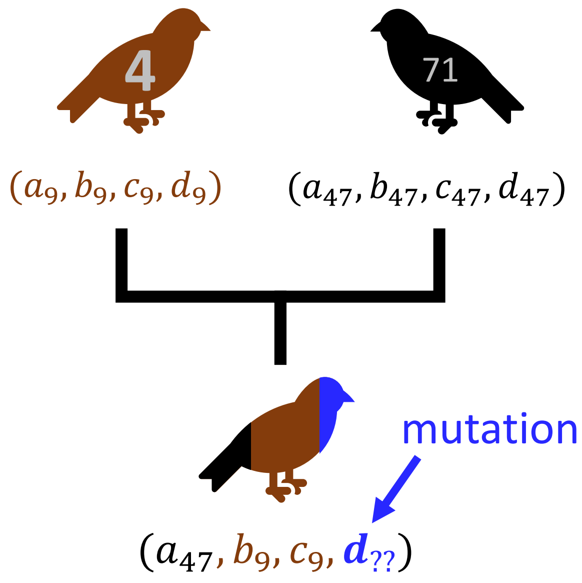
Example Algorithm #3: Evolutionary (Genetic) Algorithm

Pros

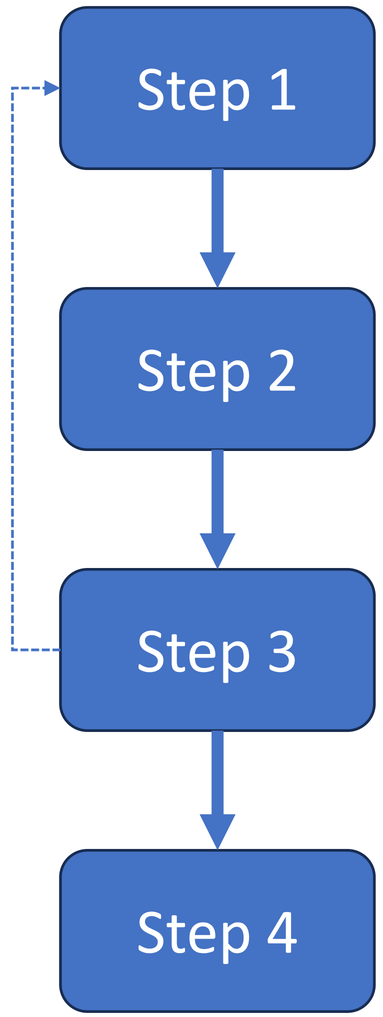
- Better at finding a global optimum than gradient descent and less computationally intensive than grid search

Cons

- Performance depends strongly on choice of parameters and algorithm
- Many choices for the parameters and algorithm, which may require time to adjust for a given optimization problem.


4
71
(a_9, b_9, c_9, d_9)
$(a_{47}, b_{47}, c_{47}, d_{47})$
mutation
$(a_{47}, b_9, c_9, \boldsymbol{d}_{??})$

Parameter optimization: Conclusion



Choose model.

- Izhikevich, Hodgkin-Huxley, etc.

Identify parameters that need to be fitted.

- E.g., perhaps membrane capacitance is constrained by data and doesn’t need to be fitted automatically.

Define loss function to compare model outcome to data

- Phase plane trajectory, inter-spike intervals, current-frequency relationship, spike adaption, sag, etc.

Choose optimization algorithm

- Grid search, gradient descent, evolutionary algorithm, etc.