WAM-BAMM '05

Parameter Searching in Neural Models

Michael Vanier
California Institute of Technology

Outline

- Defining the problem
- Issues in parameter searching
- Methodologies
  - "analytical" methods vs. stochastic methods
  - automated vs. semi-manual
- The GENESIS approach (*i.e.* my approach)
- Some results
- Conclusions and future directions

1) Defining the problem

- You have a neural model of some kind
- Your model has *parameters*
  - passive: RM, CM, RA
  - active: G_{max} of channels, $m_{inf}(V)$, tau(V), Ca dynamics
- You built it based on "best available data"
  - which means that >50% of parameters are best guesses

Defining the problem

- So you have a problem...
- How to assign "meaningful" values to those parameters?
- Want
  - values that produce correct behavior at a higher level (current clamp, voltage clamp)
  - values that are not physiologically ridiculous
  - possibly some predictive value

The old way

- Manually tweak parameters by hand
- Often large proportion of the work time of a modeler spent this way
- But we have big computers with plenty of cycles...
- Can we do better?

2) Issues in parameter searching

- How much data?
  - more data ➔ easier problem (sort of)
- How many parameters?
  - larger the parameter space, harder the problem
- Are parameter ranges constrained?
  - narrower the range, the better (usually)
- How long to simulate one "iteration" of whatever you're interested in?

Neurons vs. Networks

- All these issues compound massively with network models
- Best approach to break it down into component neurons and "freeze" neuron behaviors when wiring up network model
- Even so is very computationally intensive
  - large parameter spaces
  - long iteration times

Success criteria

- If find useful parameter set...
  - params in "reasonable" range
  - matches observable high-level data well
- ...then conclude that search has "succeeded"
- BUT:
  - Often *never* find good parameter sets
  - Not necessarily a bad thing!
  - Indicates areas where model can be improved

3) Methodologies

- Broadly speaking, two kinds of approaches:
- a) Analytical and semi-analytical approaches
  - cable theory
  - nonlinear dynamical systems (phase plane)
- b) Stochastic approaches
  - genetic algorithms, simulated annealing, etc.

Cable theory

- Can break a neuron down into compartments which (roughly) obey the cable equation
- In some cases can analytically solve for the behavior expected given certain input stimuli
- But...
  - theory only powerful for passive neurons
  - need parameters throughout a dendritic tree
  - ideally want *e.g.* 3/2 power law rule
- How helpful is this, really?

Nonlinear dynamics

- Can take any system of equations and vary e.g. two parameters and look at behavior
  - called a phase plane analysis
- Can sometimes give great insight into what is really going on in a simple model
- But...
  - assumes that all behaviors of interest can be decomposed into semi-independent sets of two parameters

Analytical approaches

- are good because they can give great insight into simple systems
- are often not useful because they are restricted to simple systems for all practical purposes
- Jim Bower: "I'd like to see anyone do a phase plane analysis of a Purkinje cell."

Analytical approaches

- In practice, users of these approach also do curve-fitting and a fair amount of manual parameter adjusting
- We want to be able to do automated parameter searches

Aside: hill-climbing approaches

- One class of automated approaches is multidimensional "hill climbing"
- AKA "gradient ascent" (or descent)
- Commonly-used method is conjugate gradient method
- We'll see more of this later

Stochastic approaches

- Hill-climbing methods tend to get stuck in local minima
- Very nonlinear systems like neural models have *lots* of local minima
- Stochastic approaches involve randomness in some fundamental way to beat this problem
  - Genetic algorithms
  - Simulated annealing
  - others

Simulated annealing

- Idea:
  - Have some way to search parameter space that works, but may get stuck in local minima
  - Run simulation, compute goodness of fit
  - Add noise to goodness of fit proportional to "temperature" which starts out high
  - Slowly reduce temperature while continuing search
  - Eventually, global maximum GOF reached

Genetic algorithms

- Idea:
  - Have a large group of different parameter sets
    - a "generation"
  - Evaluate goodness of fit for each set
  - Apply genetic operators to generation to create next generation
    - fitness-proportional reproduction
    - mutation
    - crossing-over

Genetic algorithms (2)

- Crossing over is slightly weird
- Take part of one param set and splice it to rest of another param set
  - many variations
- Works well if parameter "genome" is comprised of many semi-independent groups
- Therefore, order of parameters in param set matters!
  - *e.g.* put all params for a given channel together

Questions

- Which methods work best?
- And under which conditions?
  - passive vs. active models
  - small # of params vs. large # of params
  - neurons vs. networks

Parameter searching in GENESIS

- I built a GENESIS library to answer these questions
  - and for my own modeling efforts
  - and to get a cool paper out of it
- Various parameter search "objects" in param library

The param library

- GENESIS objects:
  - paramtableBF: brute force
  - paramtableCG: conjugate gradient search
  - paramtableSA: simulated annealing
  - paramtableGA: genetic algorithms
  - paramtableSS: stochastic search

How it works

- You define your simulation
- You specify what "goodness of fit" means
  - waveform matching
  - spike matching
  - other?
- You define what your parameters are
  - load this info into paramtableXX object
- Write simple script function(s) to
  - run simulation
  - update parameters
- Until acceptable match achieved

How it works

- Scripts library of genesis contains demos for all paramtable objects
- Easiest way to learn
- I'll walk you through it later if you want

Some results

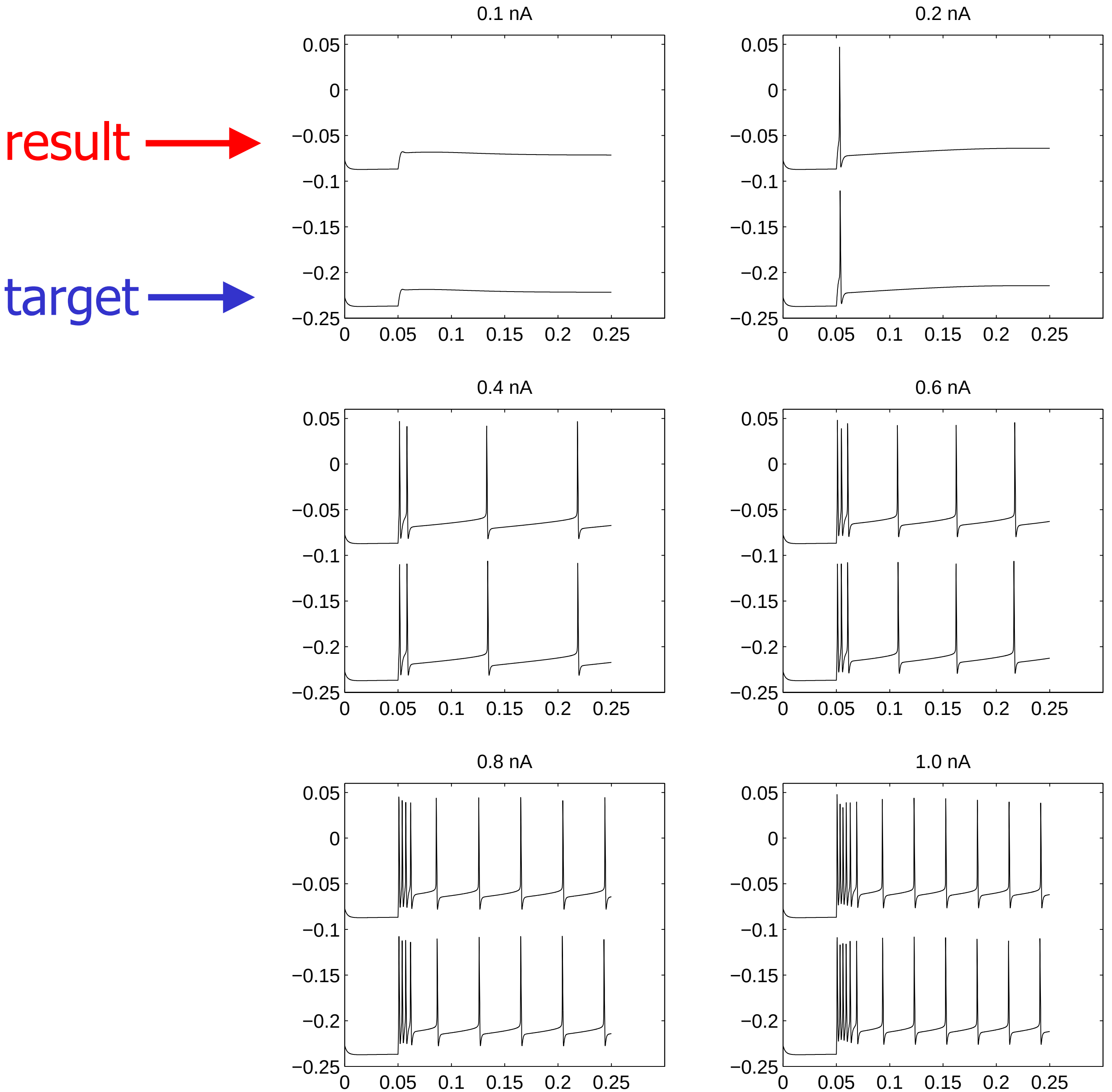
- Models:
  - active 1-compartment model w/4 channels
    - 4 parameters (G_{max} of all channels)
    - 8 parameters (G_{max} and tau(V) of all channels)
  - linear passive model w/ 100 compartments
    - params: RM, CM, RA
  - passive model w/ 4 dendrites of varying sizes
    - params: RM, CM, RA of all dendrites + soma
  - pyramidal neuron model w/15 compartments, active channels (23 params of various types)

Goodness of fit functions

- For passive models, match waveforms pointwise
- For active models, cheaper to match just spike times
  - hope that interspike waveforms also match
  - test of predictive power of approach

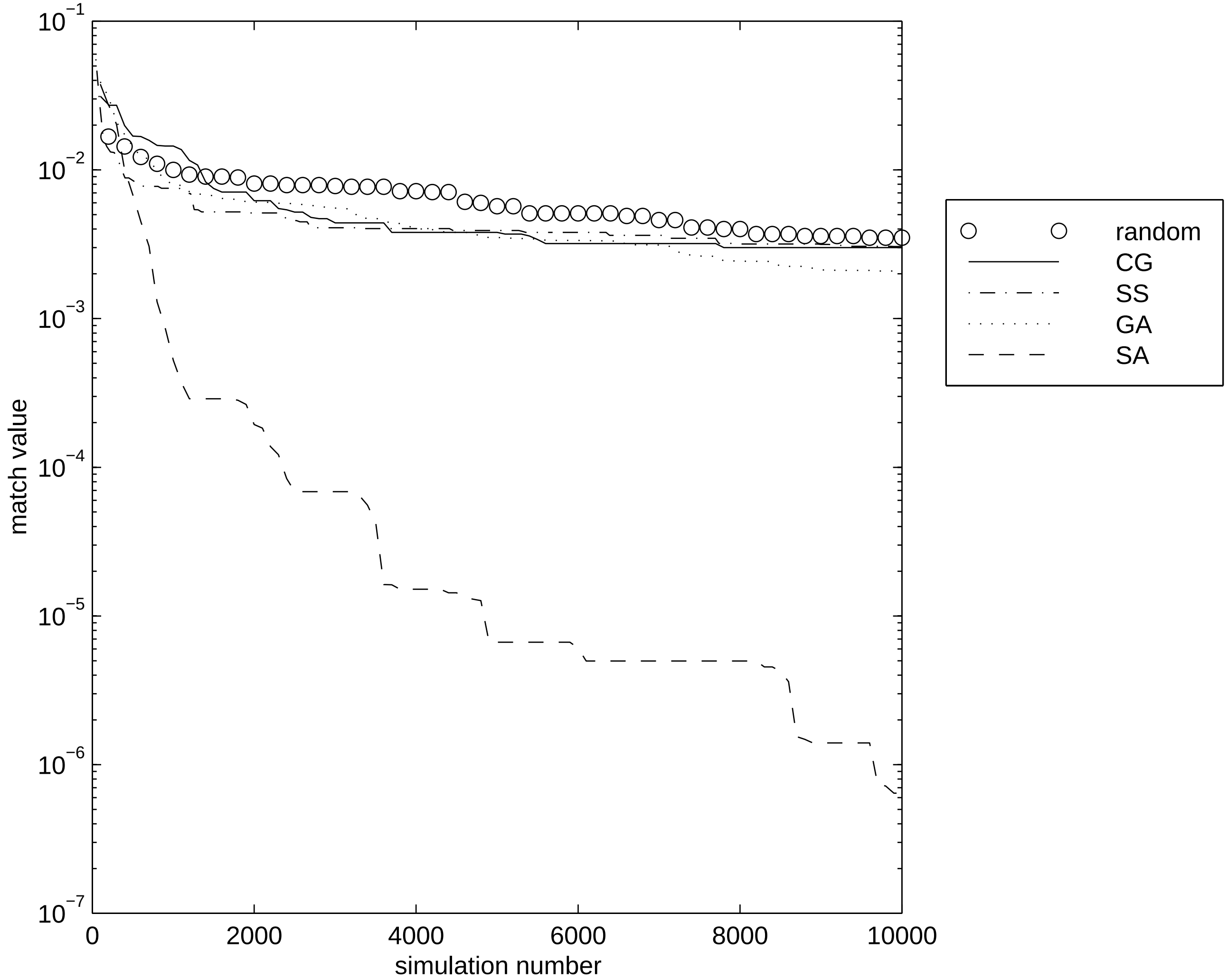
Results for 1-compartment model

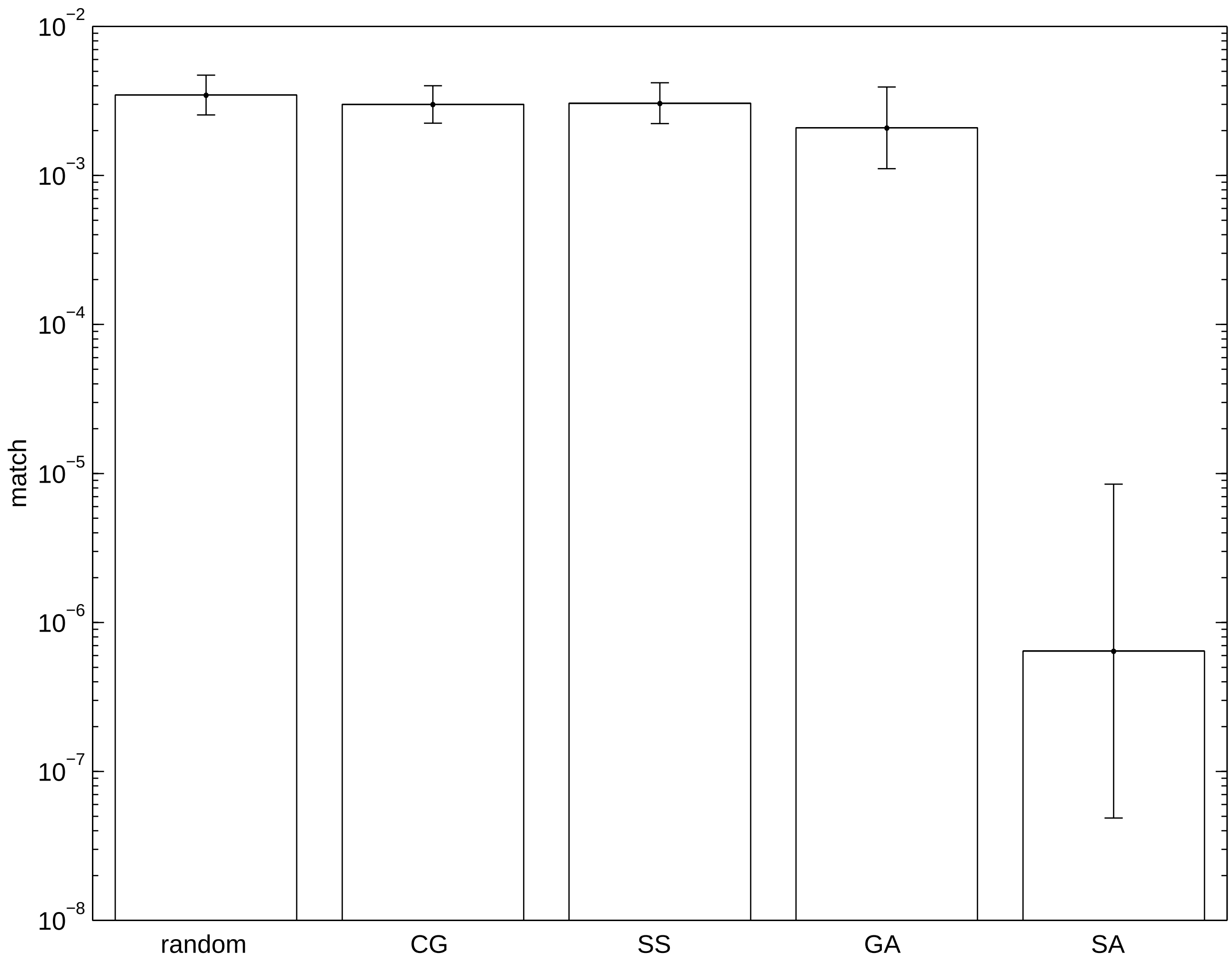
- Upper traces represent results from model found by param search
- Lower traces represent target data
- Target data offset by -150 mV for clarity
- Each trace represents a separate level of current injection
- Resolution of figures is poor
  - blame Microsoft



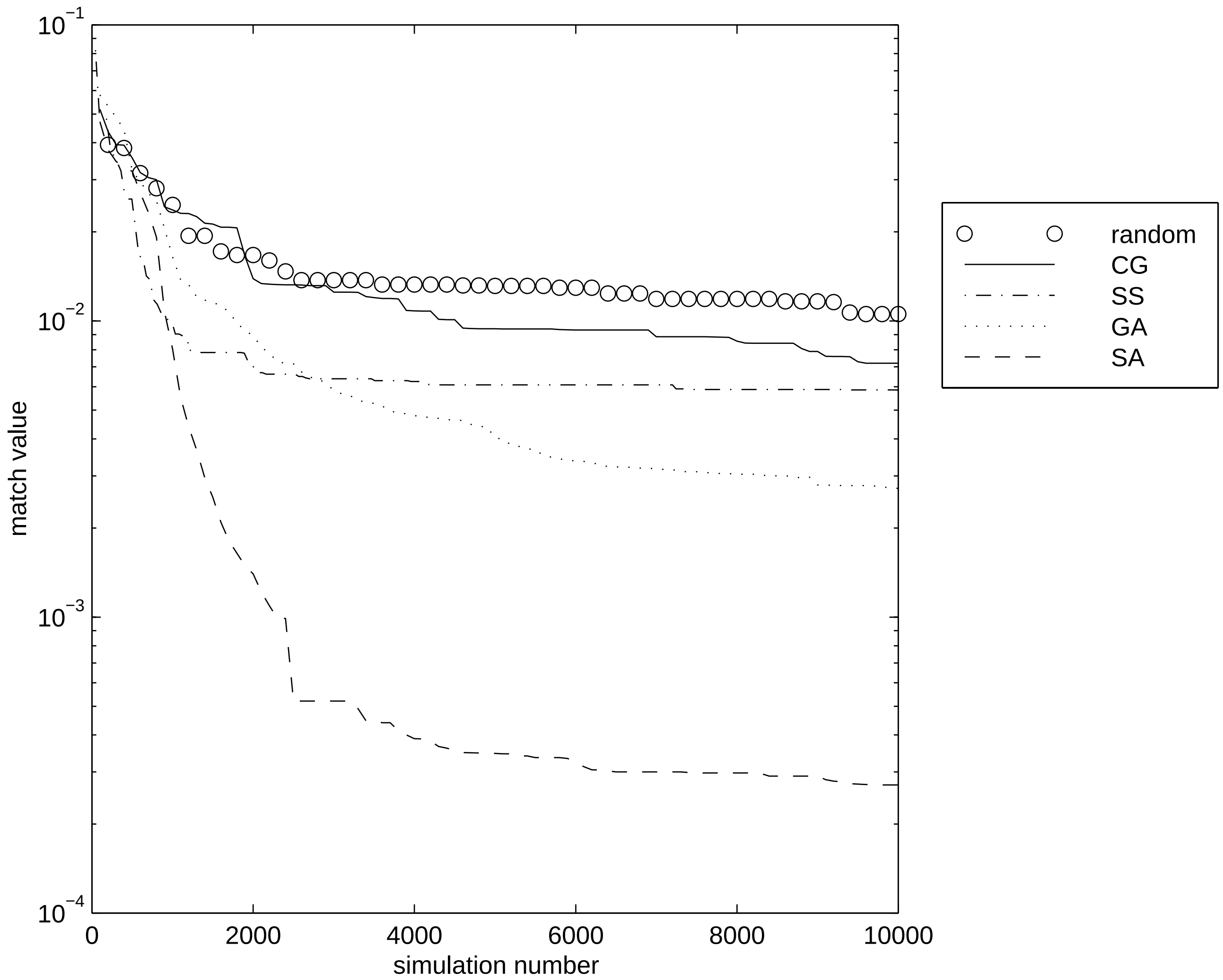
result
target
0.1 nA
0.2 nA
0.4 nA
0.6 nA
0.8 nA
1.0 nA
0.05
0
−0.05
−0.1
−0.15
−0.2
−0.25
0 0.05 0.1 0.15 0.2 0.25

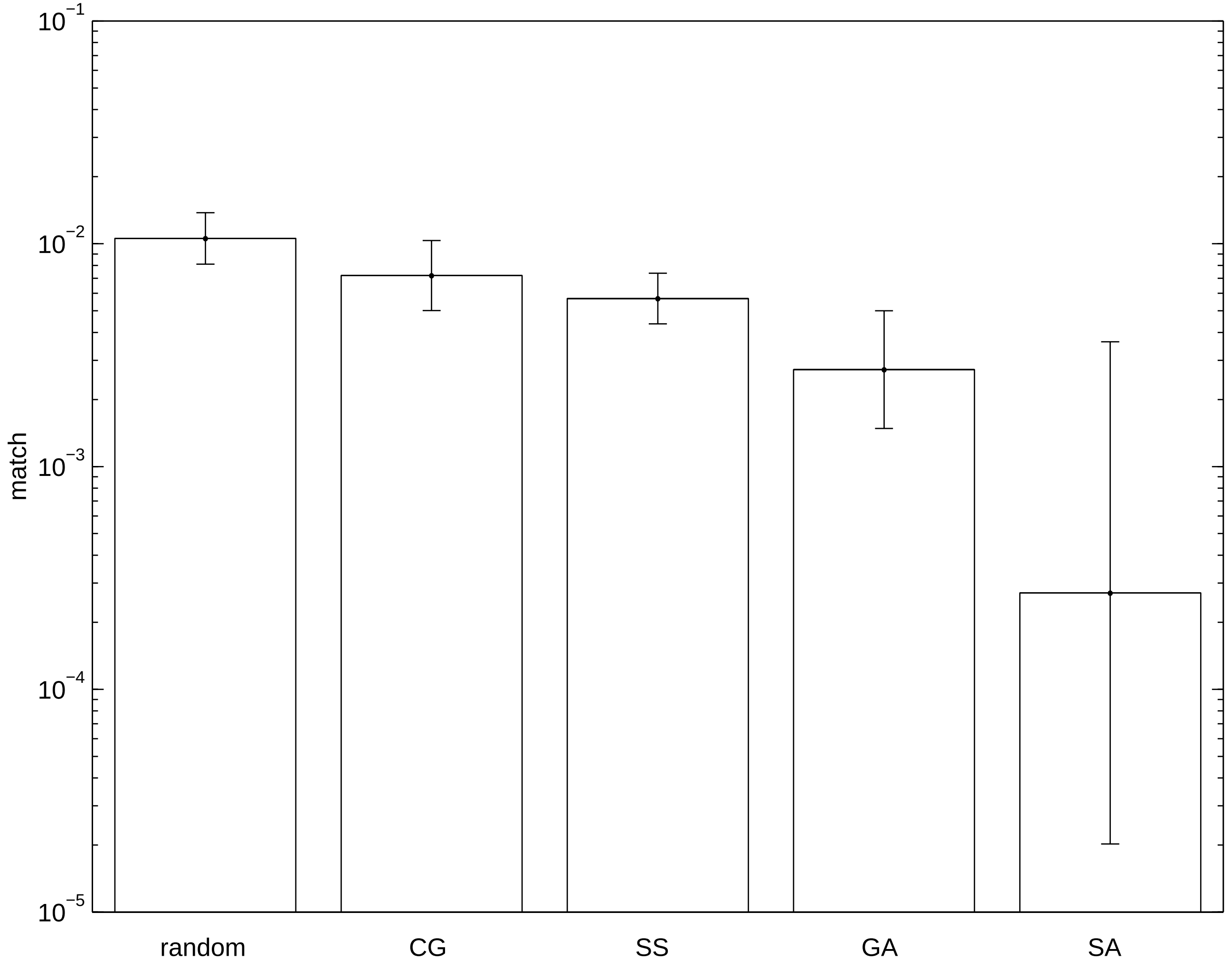
Results for 1-compt model (4 params)




match
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
10^{-8}
random
CG
SS
GA
SA

Results for 1-compt model (8 params)





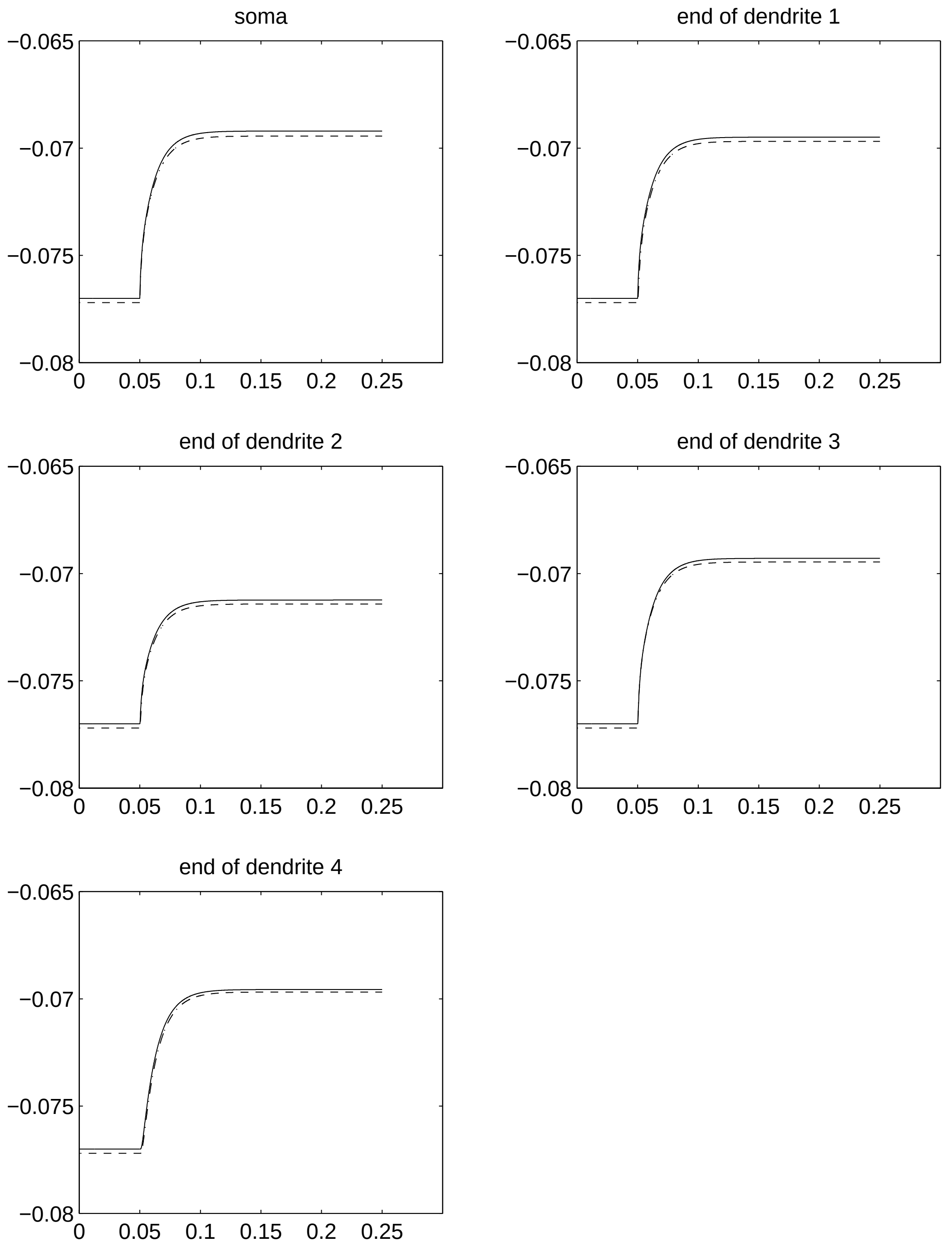
10^−1
10^−2
10^−3
10^−4
10^−5
match
random
CG
SS
GA
SA

1-compt models: conclusions

- 4 parameter model: SA blows away the competition
- 8 parameter model: SA best, GA also pretty good

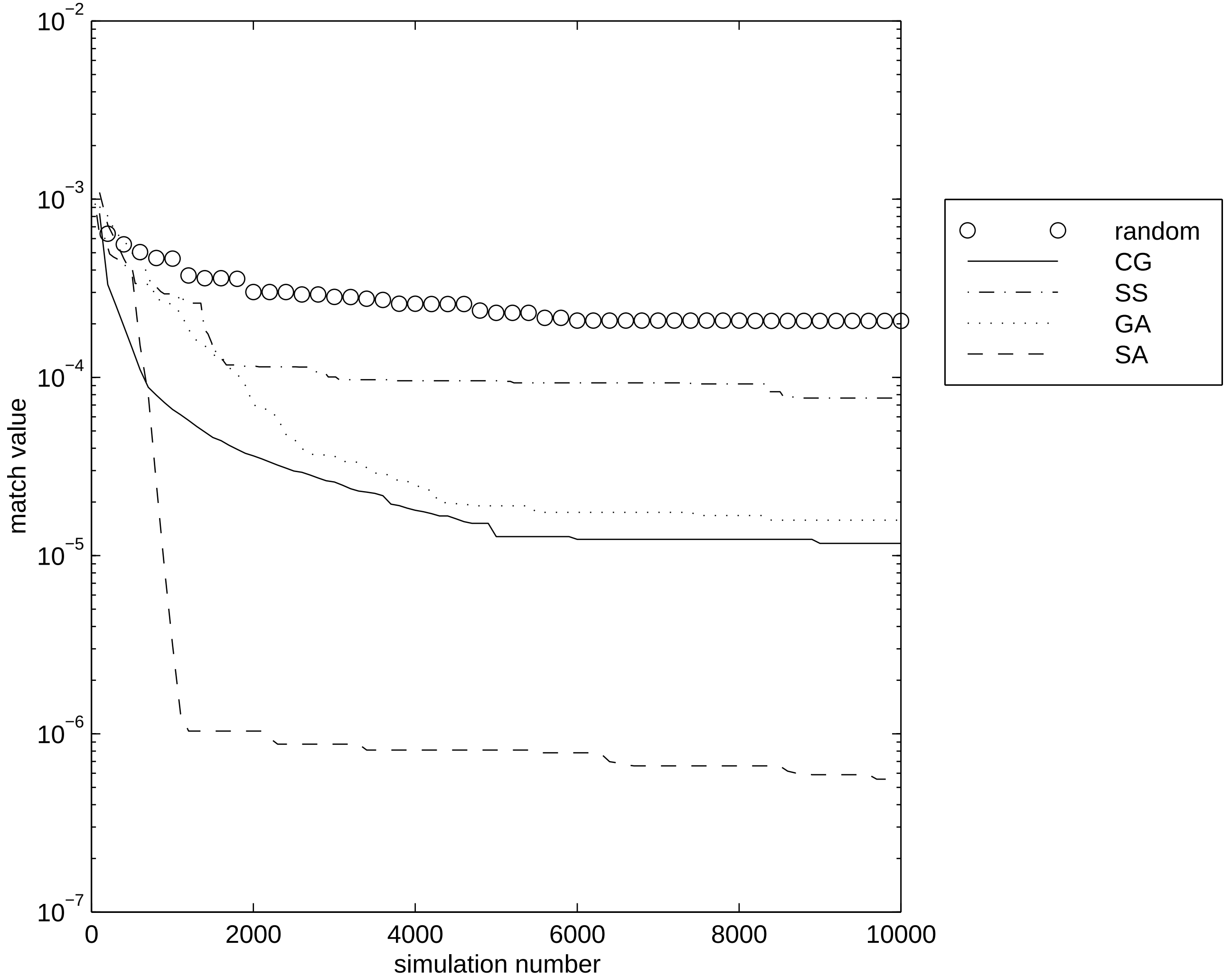
Results for passive models

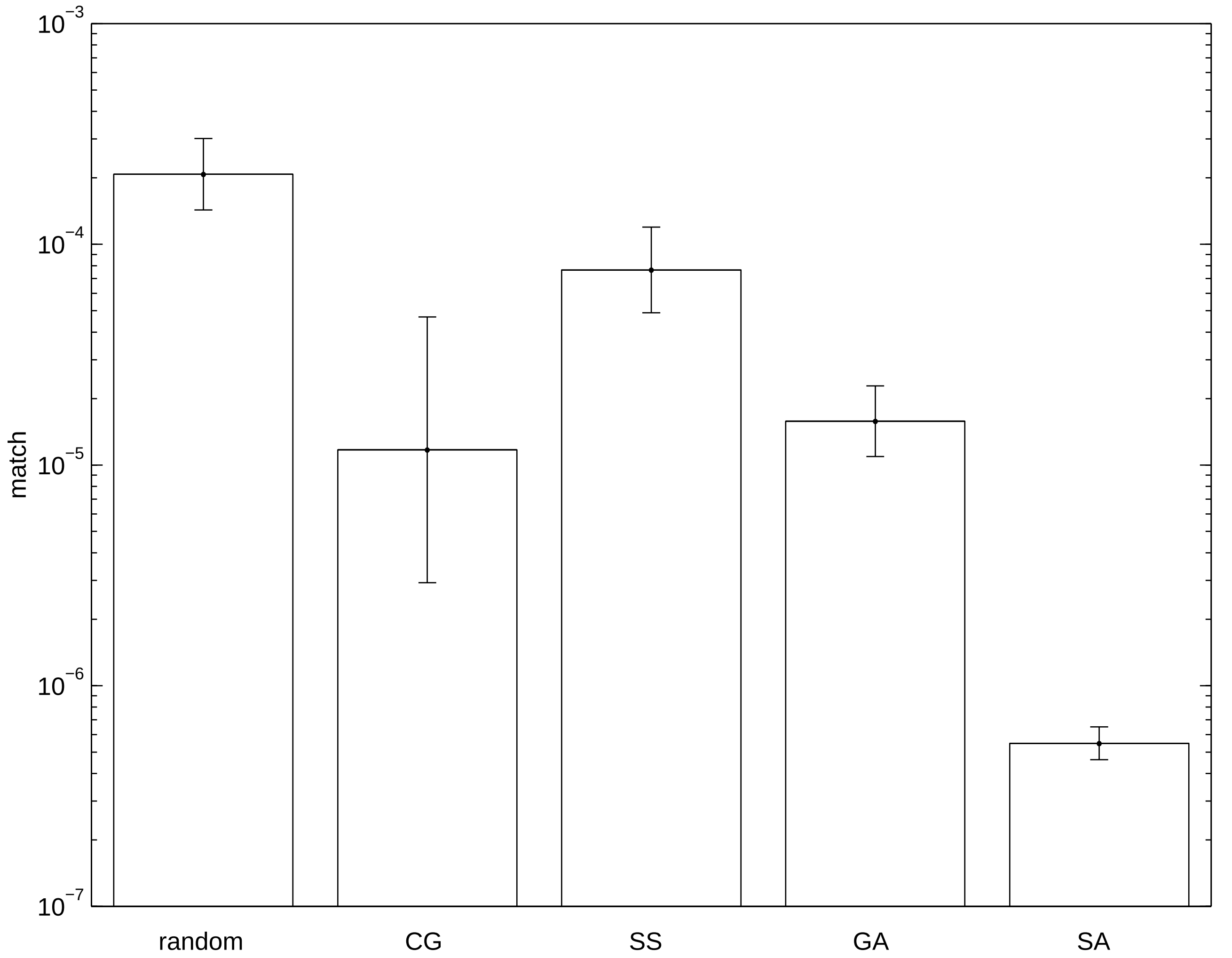
- Solid lines represent results from model found by param search
- Broken lines represent target data
- Target data offset by -2 mV
  - otherwise would overlap completely



soma
end of dendrite 1
end of dendrite 2
end of dendrite 3
end of dendrite 4
−0.065
−0.07
−0.075
−0.08
0
0.05
0.1
0.15
0.2
0.25

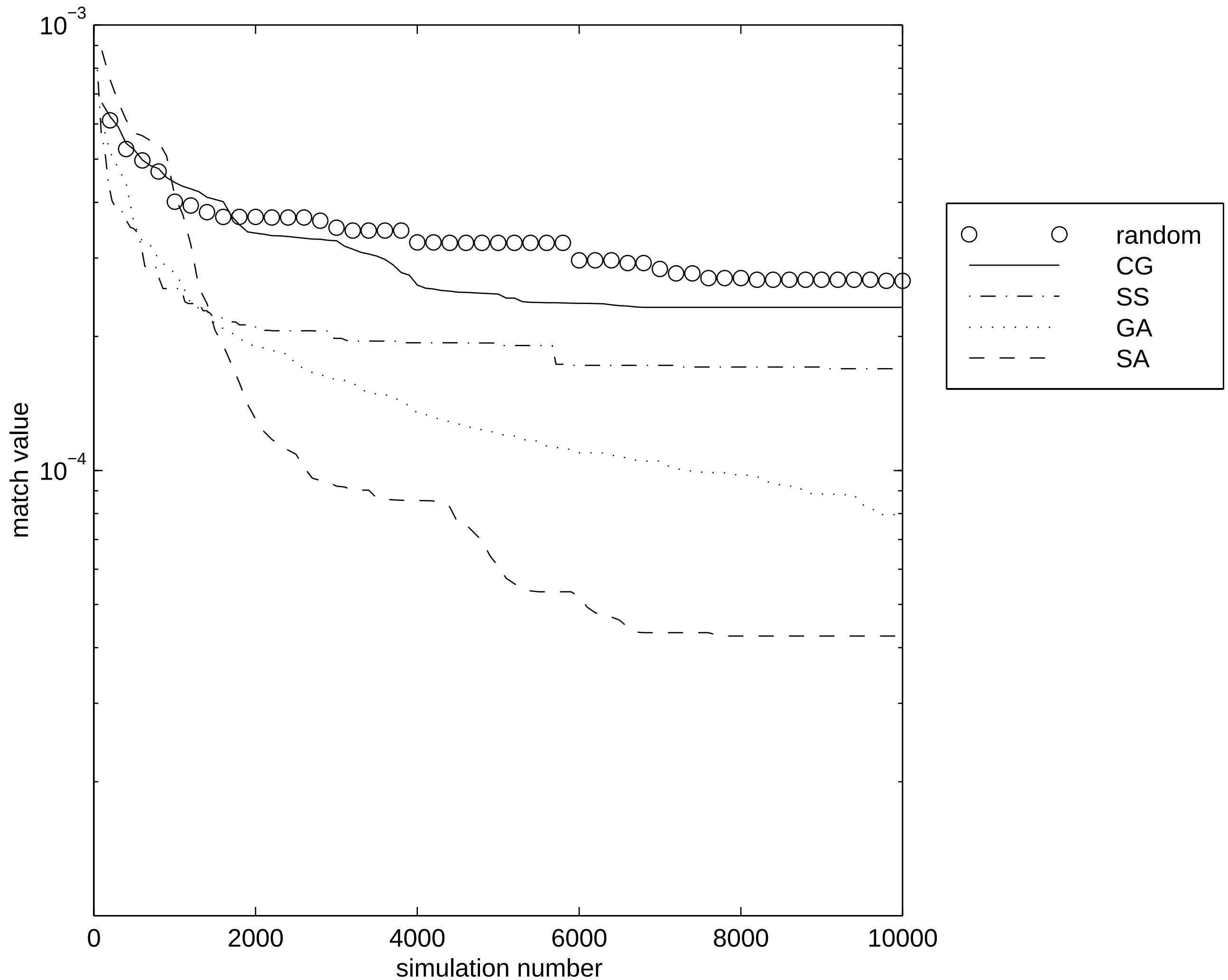
Results for passive model 1

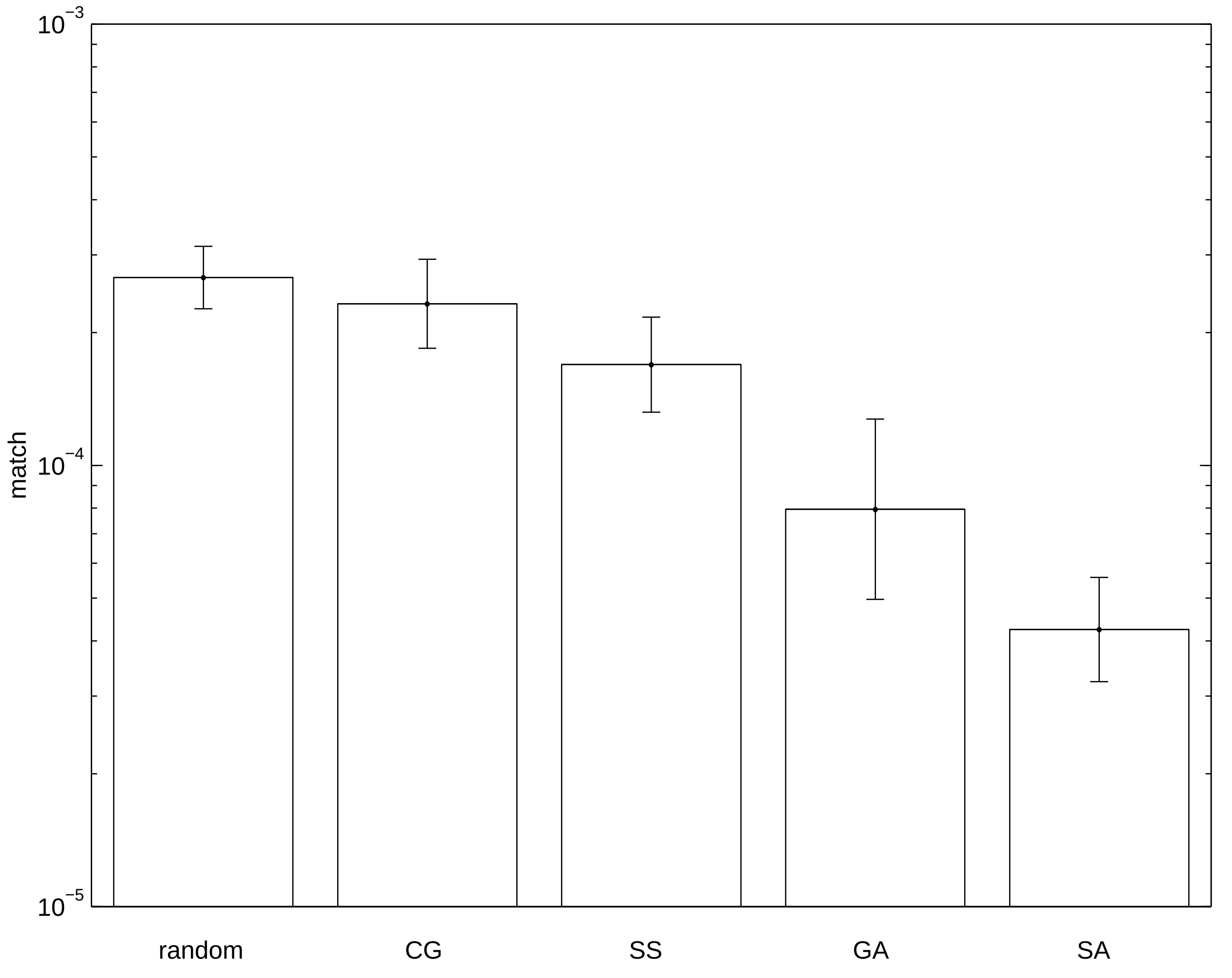




10^{-3}
10^{-4}
10^{-5}
10^{-6}
10^{-7}
match
random
CG
SS
GA
SA

Results for passive model 2





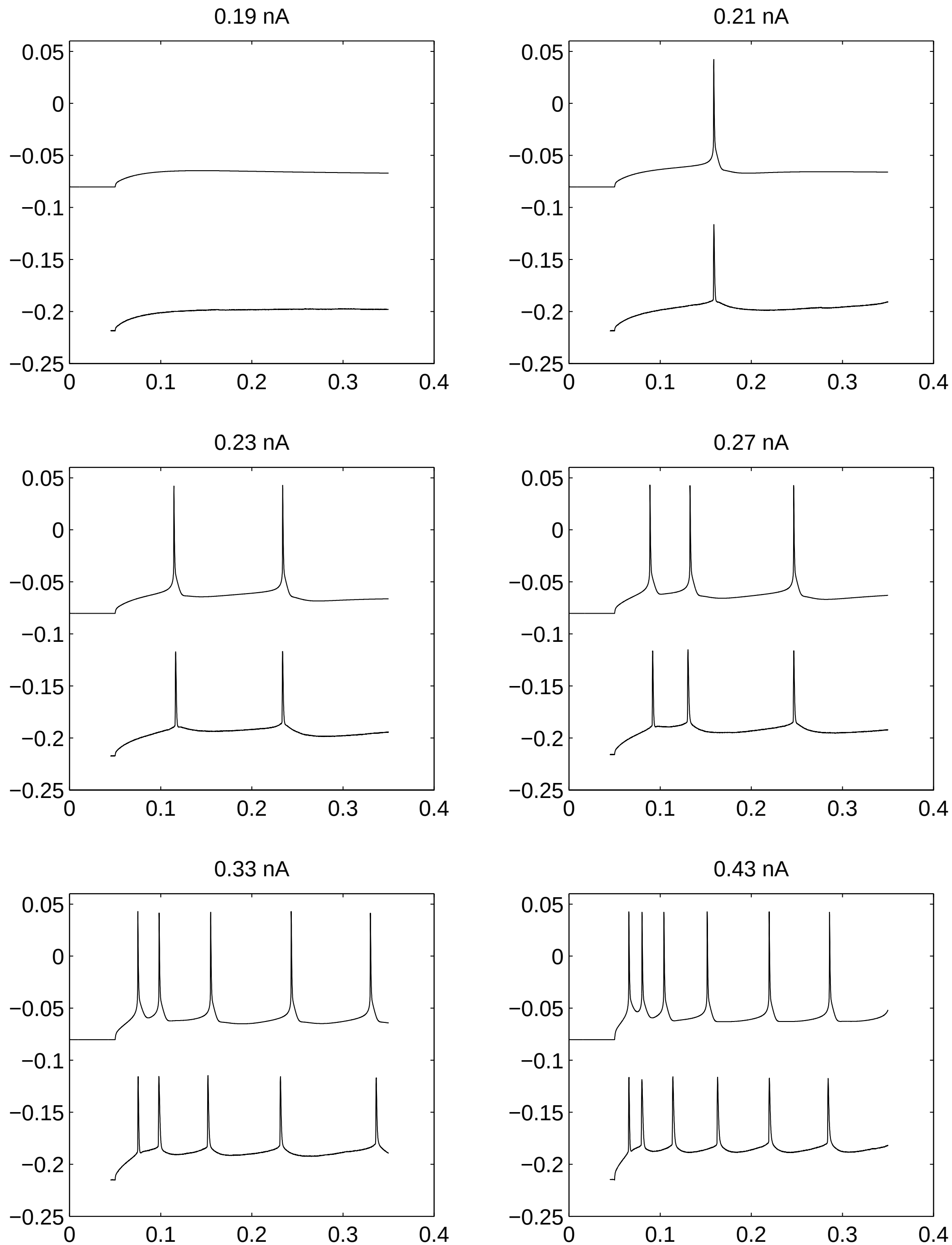
10^{-3}
10^{-4}
10^{-5}
match
random
CG
SS
GA
SA

Passive models: conclusions

- 3 parameter model:
  - SA still best
  - CG does surprisingly well
- 15 parameter model:
  - SA again does best
  - GA now strong second

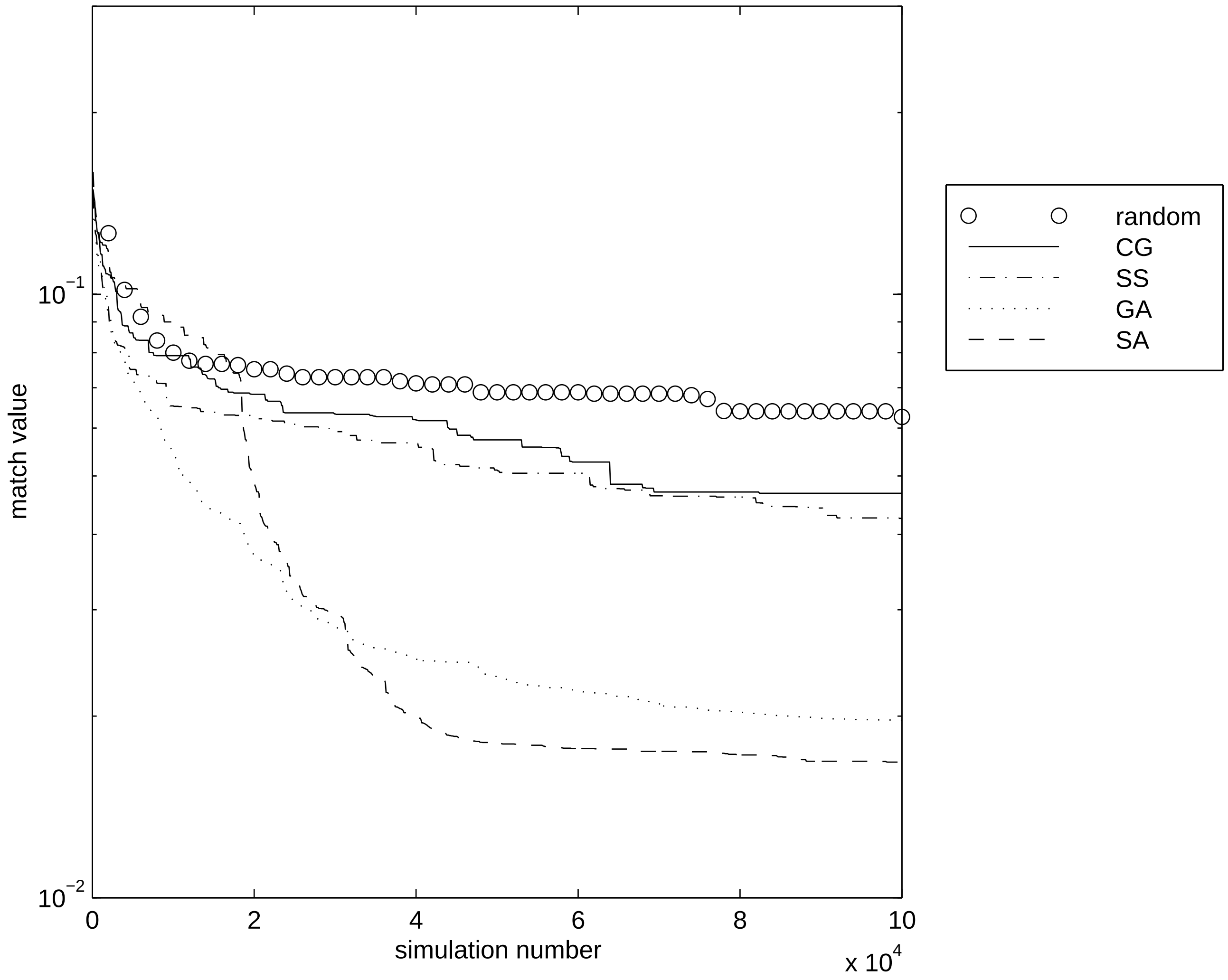
Results for pyramidal model

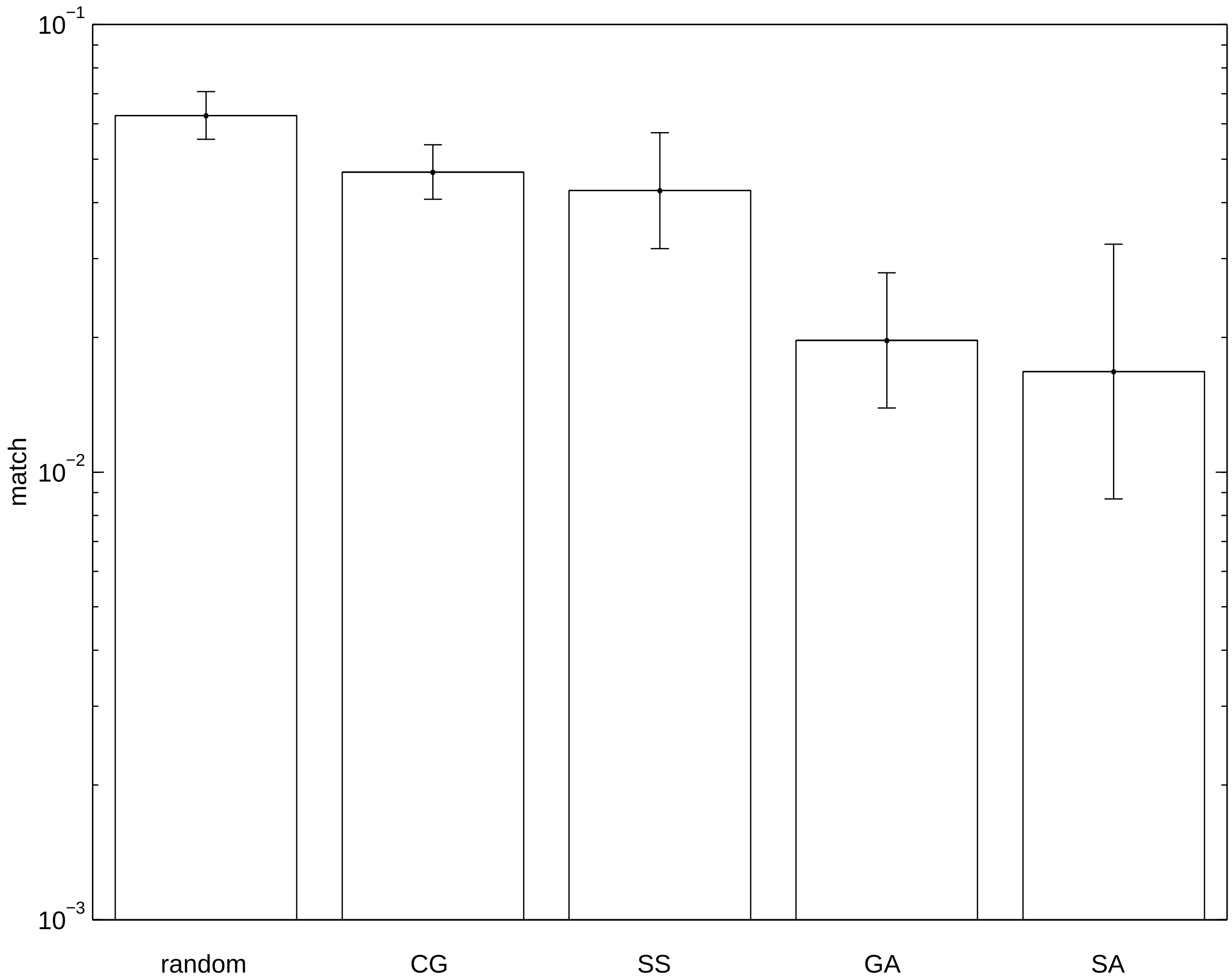
- Upper traces represent results from model found by param search
- Lower traces represent experimental data
- Experimental data offset by -150 mV for clarity



0.19 nA
0.21 nA
0.23 nA
0.27 nA
0.33 nA
0.43 nA
0.05
0
−0.05
−0.1
−0.15
−0.2
−0.25
0
0.1
0.2
0.3
0.4

Results for pyramidal neuron model





10^{-1}
10^{-2}
10^{-3}
match
random
CG
SS
GA
SA

Pyramidal model: conclusions

- Spike times matched extremely well
- Interspike waveform less so, but still reasonable
- SA still did best, but GA did almost as well
- Other methods not competitive

Overall conclusions

- Non-stochastic methods not competitive except for simple passive models
  - probably few local minima in those
- For small # of params, SA unbeatable
- As parameter number increases, GA starts to overtake SA
  - but problem gets much harder regardless

Caveats

- Small number of experiments
- All search methods have variations
  - especially GAs!
- We expect overall trend to hold up
  - but can't prove without more work

Possible future directions

- Better stochastic methods
  - *e.g.* merge GA/SA ideas
  - for instance, GA mutation rate that drops as function of "temperature"
  - other "adaptive SA" methods exist
- Extension to network models?
  - May now have computational power to attempt this
  - Will stochastic methods be dominant in this domain too?

Finally...

- Working on parameter search methods is fun
- Nice to be away from computer while still feeling that you're doing work
- Nice to be able to use all spare CPU cycles
- Good results feel like "magic"
- Probably a few good PhDs in this