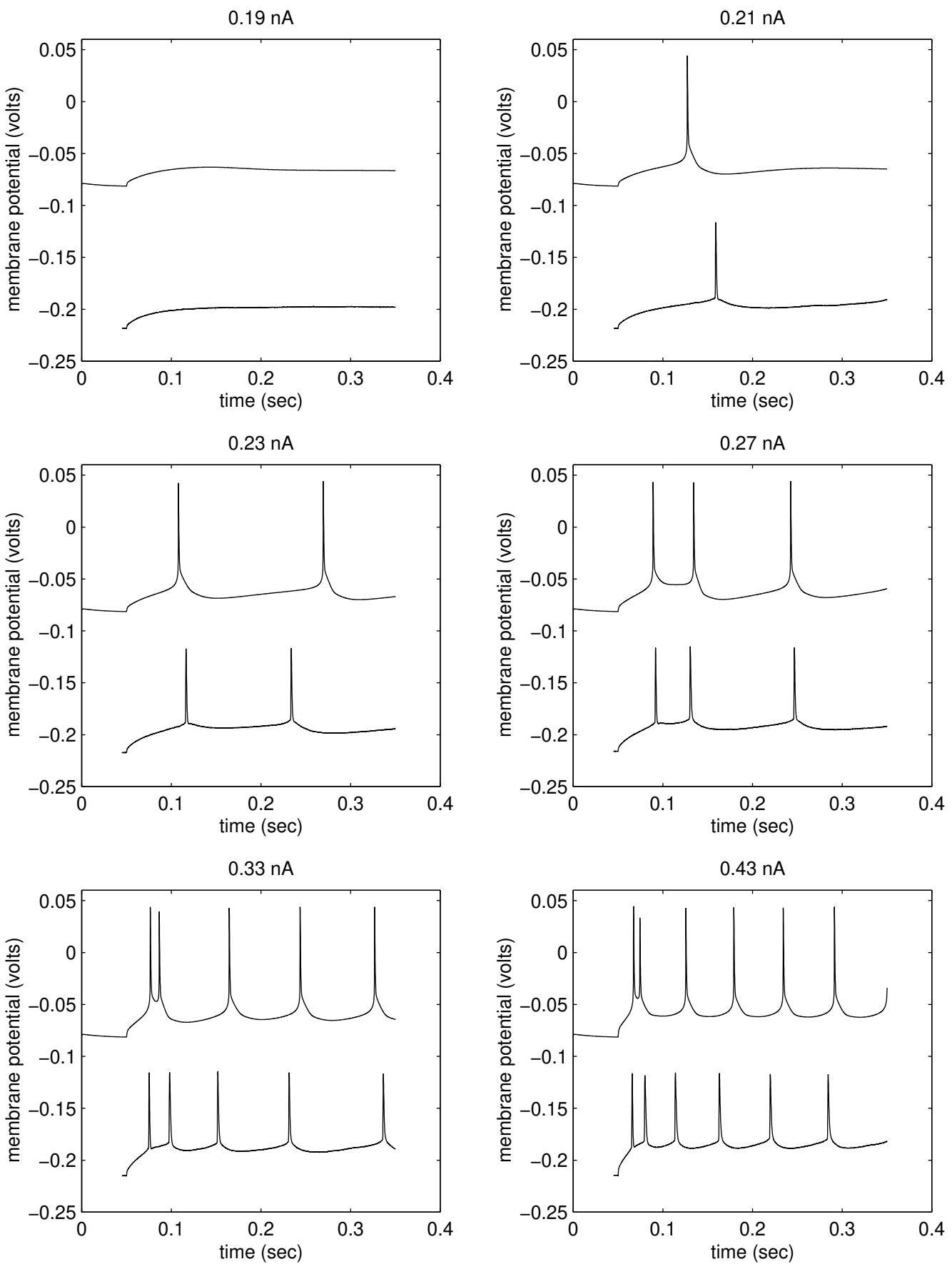
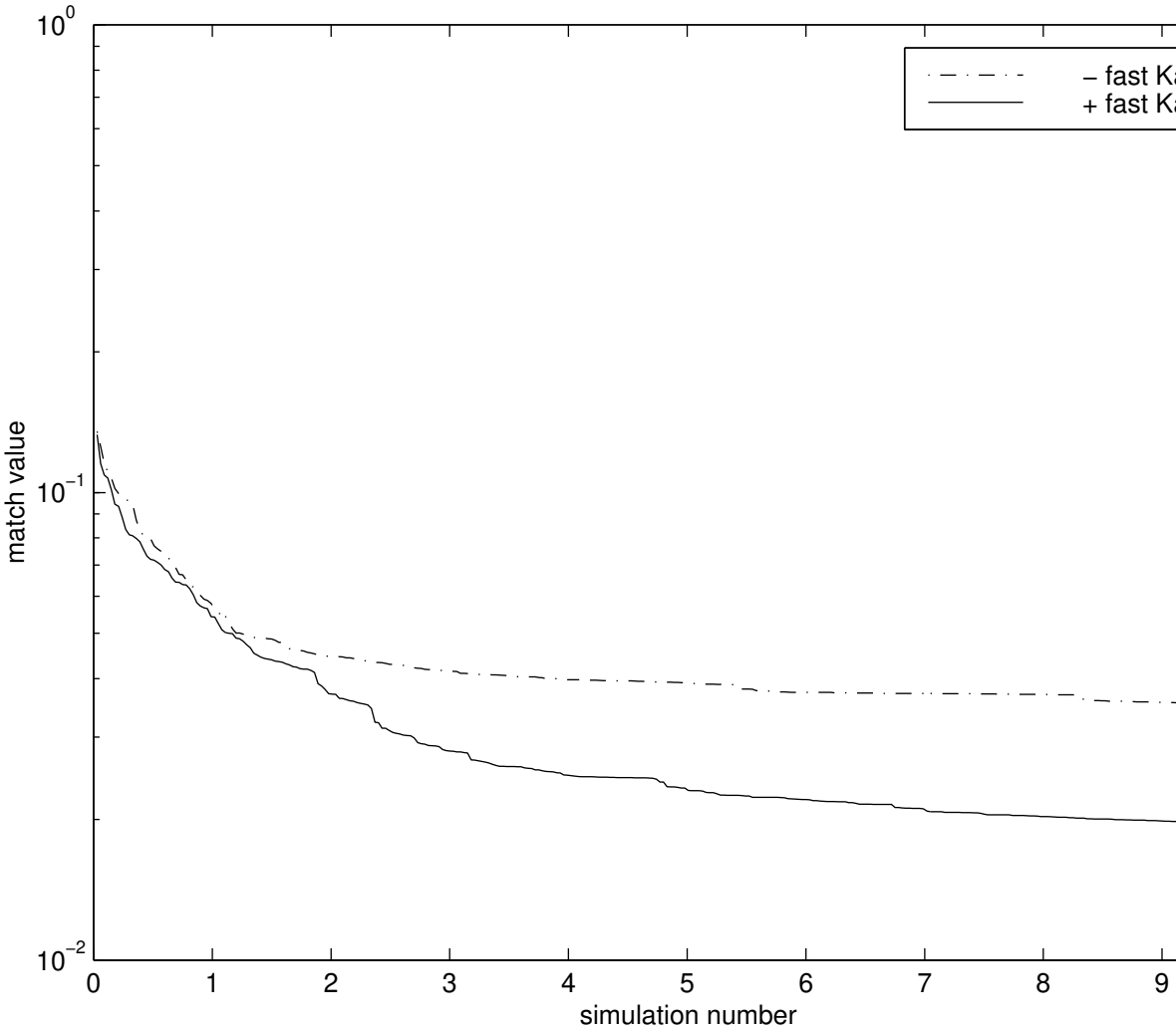
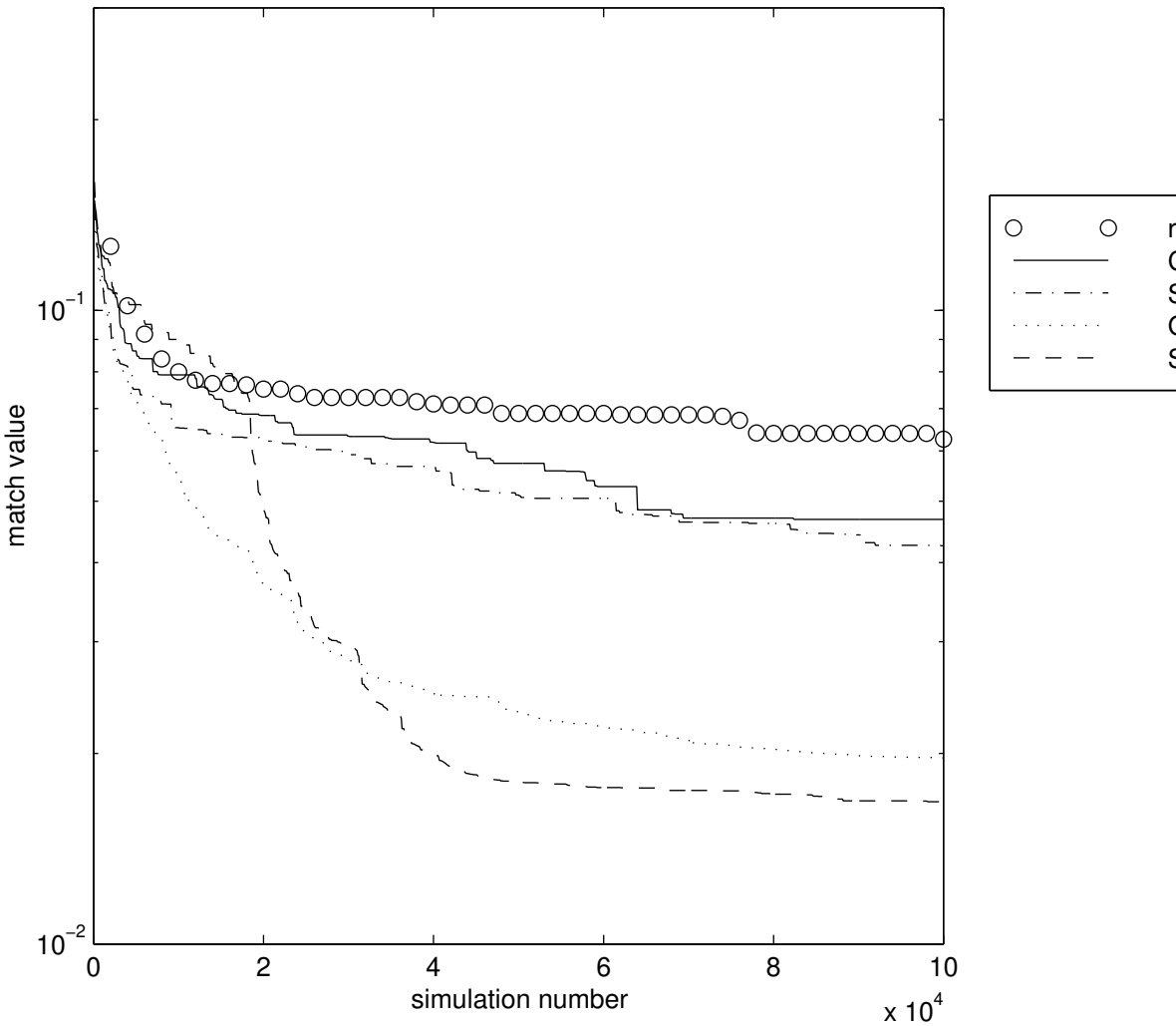
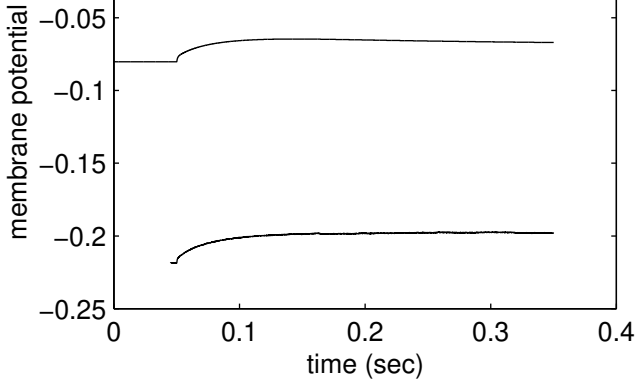
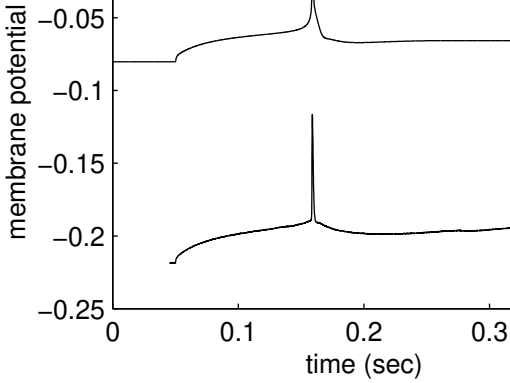

0.19 nA
0.21 nA
0.23 nA
0.27 nA
0.33 nA
0.43 nA
membrane potential (volts)
time (sec)
0.05
0
−0.05
−0.1
−0.15
−0.2
−0.25
0
0.1
0.2
0.3
0.4

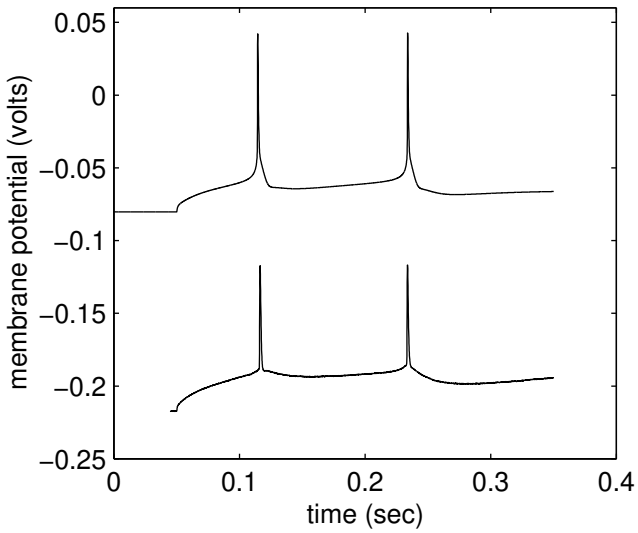

match value
simulation number
– fast K
+ fast K


match value
10^{-1}
10^{-2}
0
2
4
6
8
10
simulation number
x 10^4

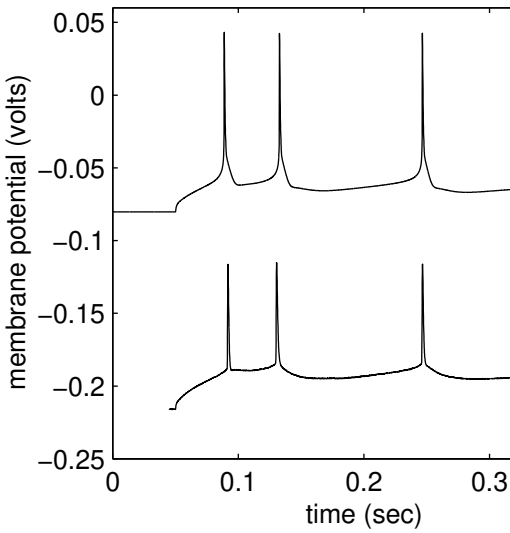




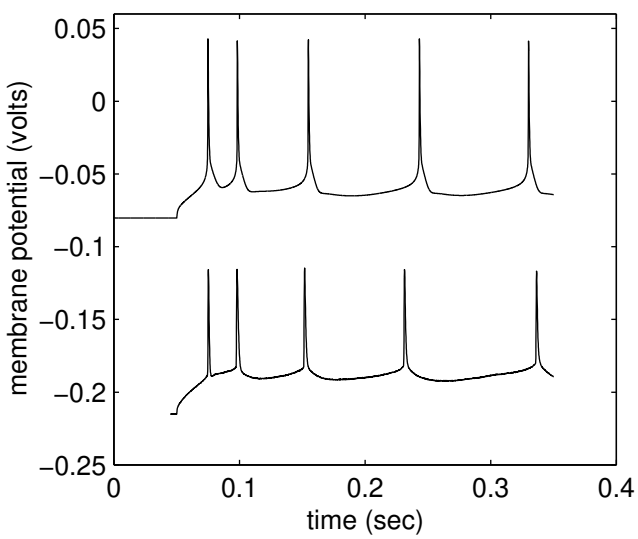
0.23 nA



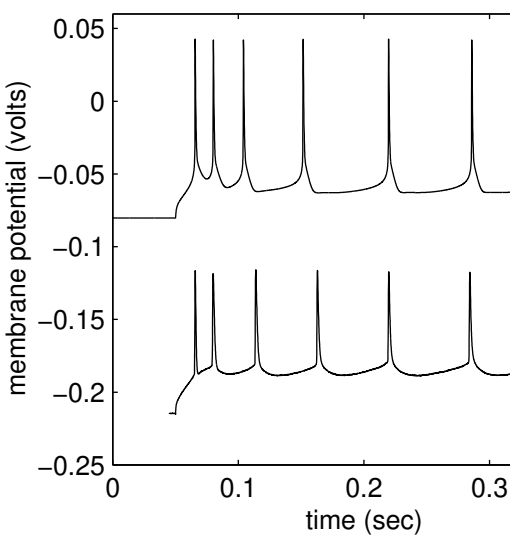
0.27 nA

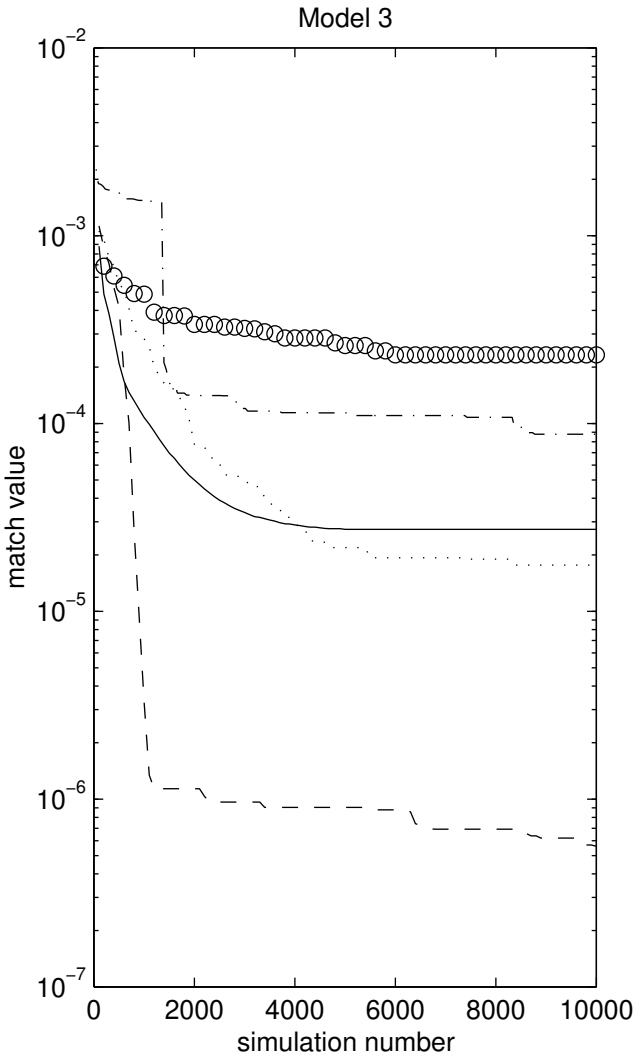


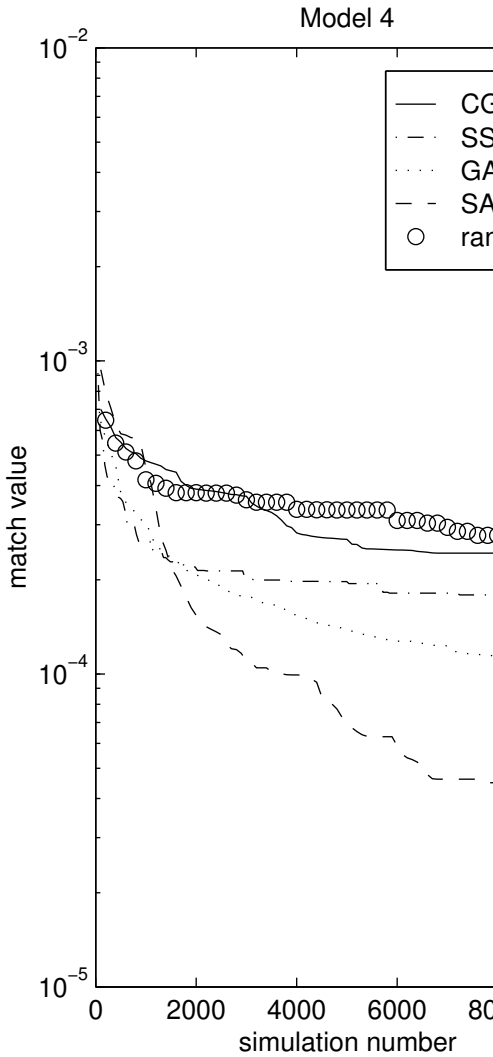
0.33 nA

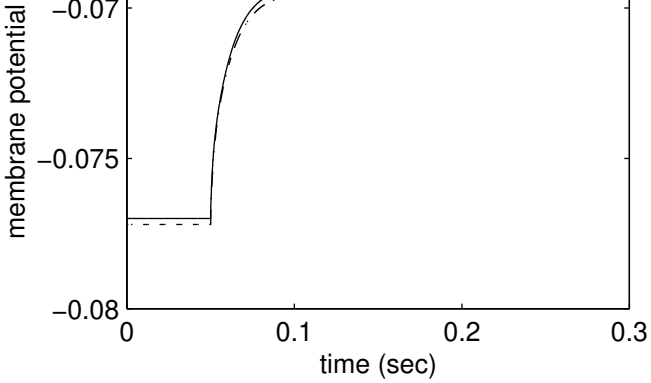


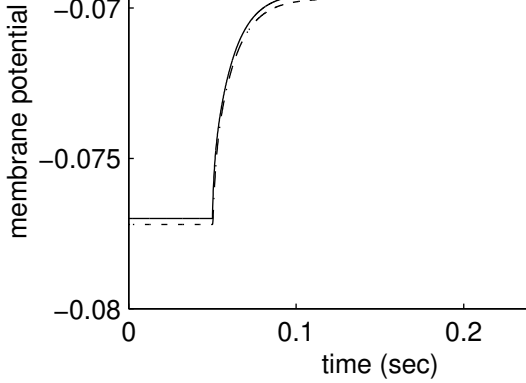
0.43 nA

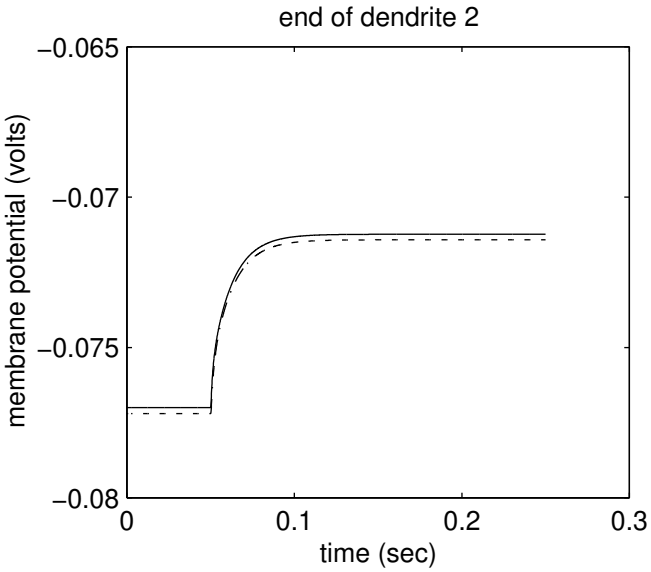


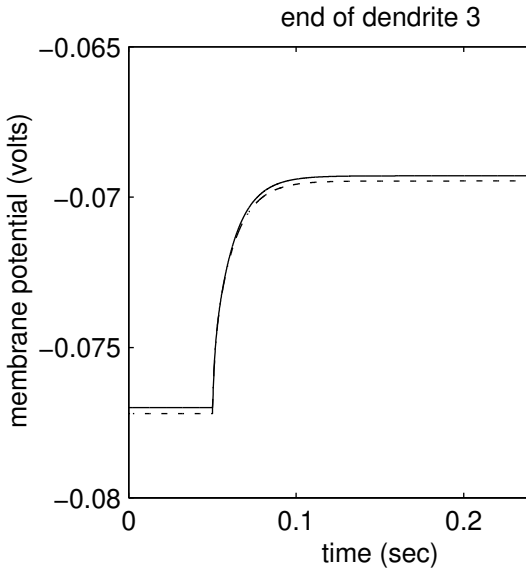

Model 3
match value
simulation number
0
2000
4000
6000
8000
10000

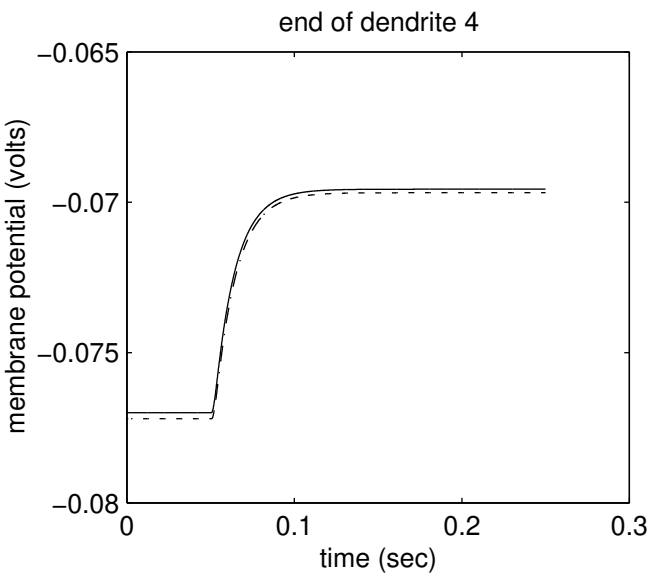

Model 4
match value
simulation number
0
2000
4000
6000
CG
SS
GA
SA
ran

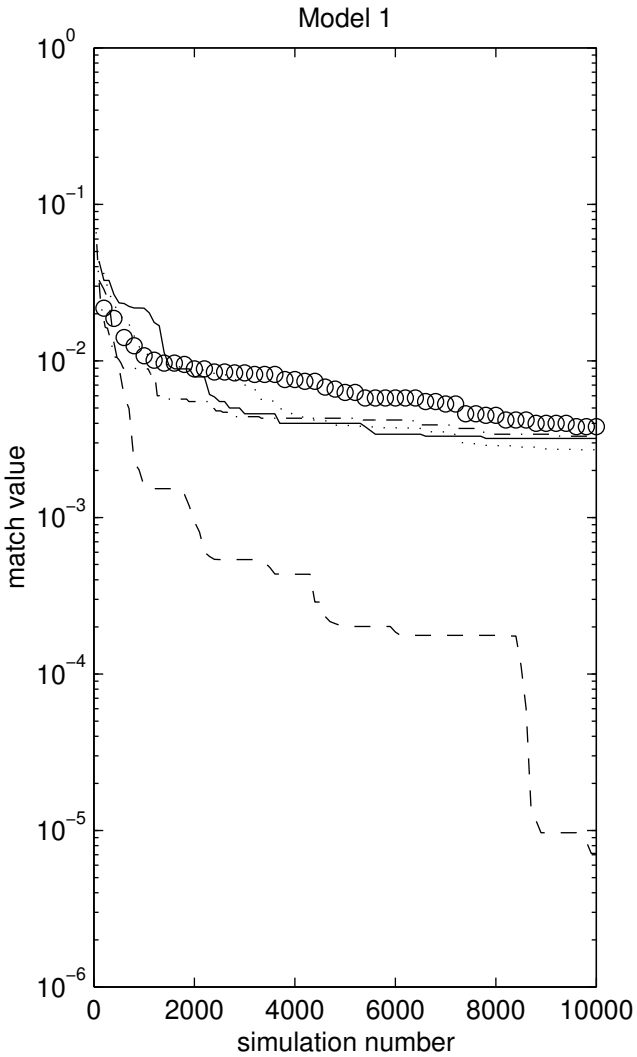

membrane potential
−0.07
−0.075
−0.08
0
0.1
0.2
0.3
time (sec)

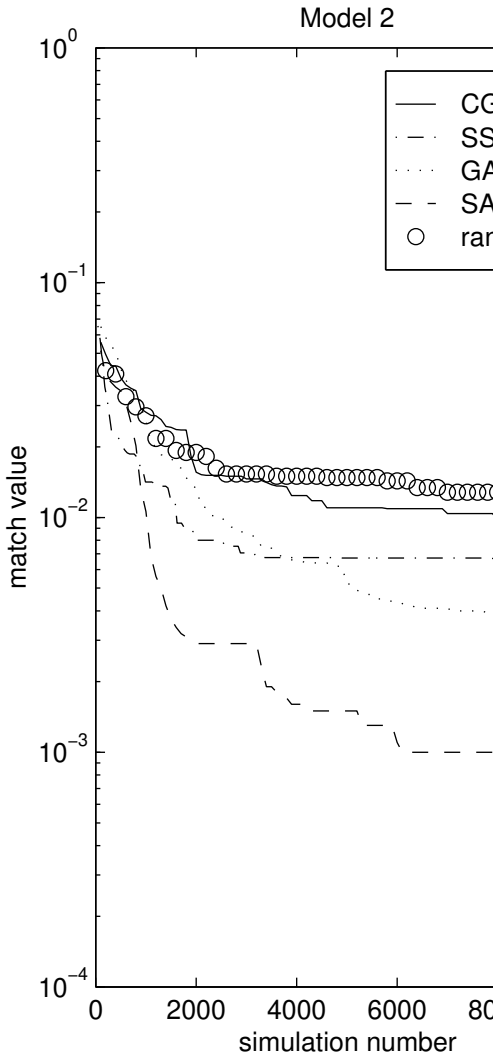

membrane potential
−0.07
−0.075
−0.08
0
0.1
0.2
time (sec)

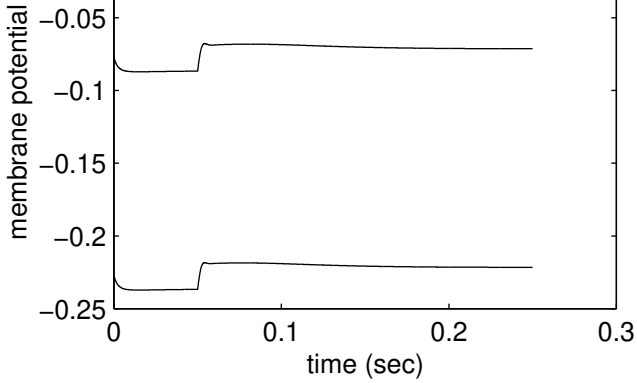

end of dendrite 2
membrane potential (volts)
−0.065
−0.07
−0.075
−0.08
0
0.1
0.2
0.3
time (sec)

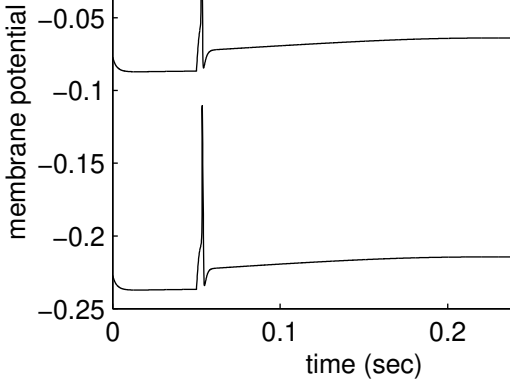

end of dendrite 3
membrane potential (volts)
−0.065
−0.07
−0.075
−0.08
0
0.1
0.2
time (sec)

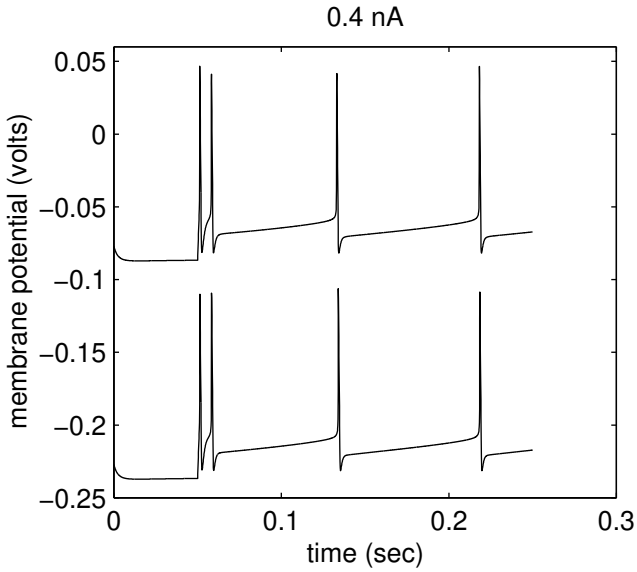

end of dendrite 4
membrane potential (volts)
−0.065
−0.07
−0.075
−0.08
0
0.1
0.2
0.3
time (sec)

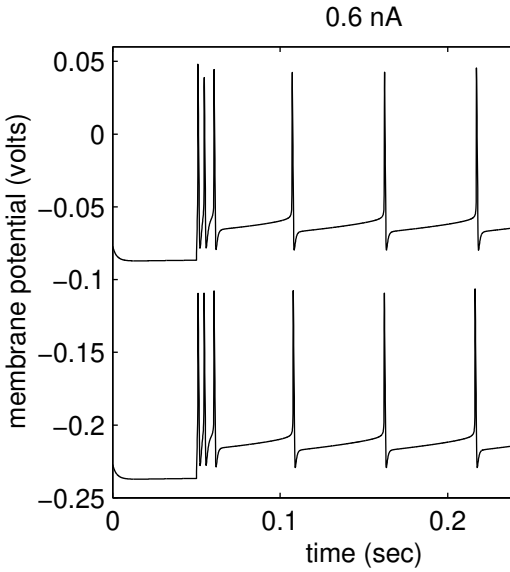

Model 1
match value
simulation number
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
10^{-5}
10^{-6}
0
2000
4000
6000
8000
10000

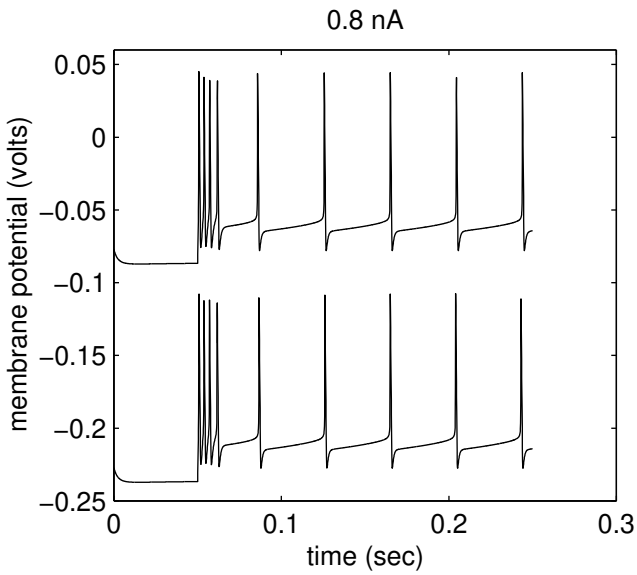

Model 2
match value
simulation number
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{-4}
0
2000
4000
6000
CG
SS
GA
SA
ran

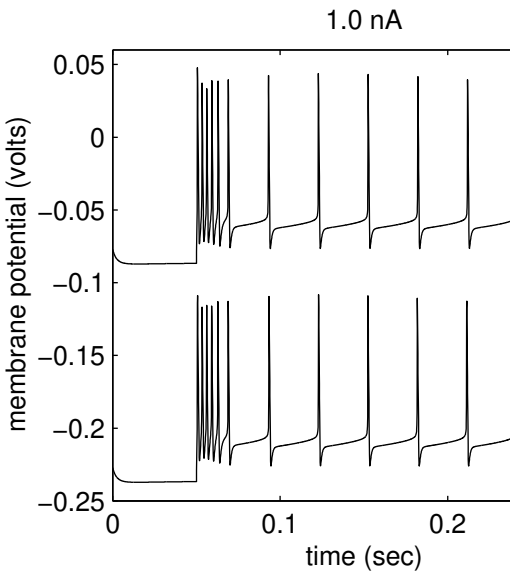

membrane potential
−0.05
−0.1
−0.15
−0.2
−0.25
0
0.1
0.2
0.3
time (sec)

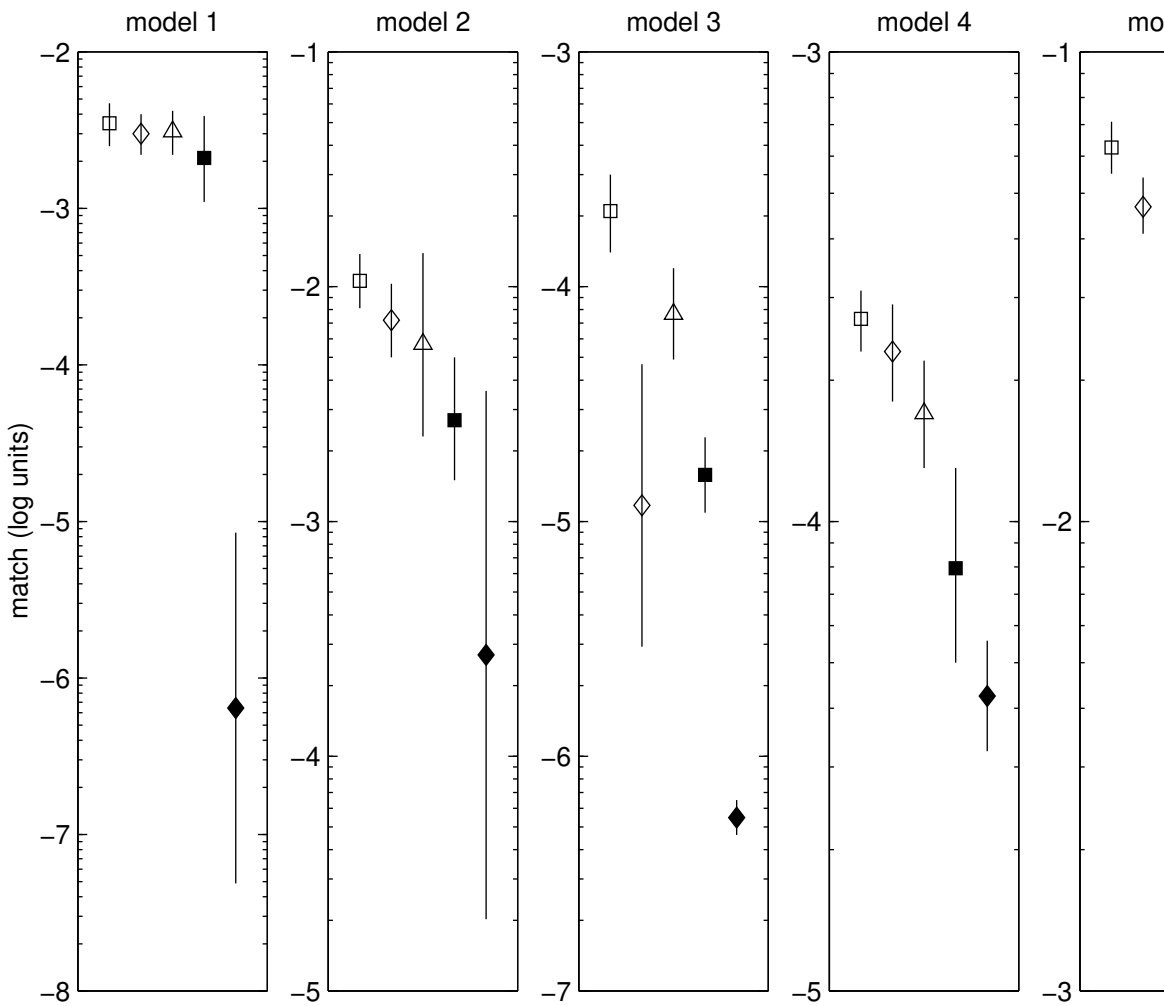

membrane potential
−0.05
−0.1
−0.15
−0.2
−0.25
0
0.1
0.2
time (sec)


0.4 nA
membrane potential (volts)
0.05
0
−0.05
−0.1
−0.15
−0.2
−0.25
0
0.1
0.2
0.3
time (sec)


0.6 nA
membrane potential (volts)
0.05
0
−0.05
−0.1
−0.15
−0.2
−0.25
0
0.1
0.2
time (sec)


0.8 nA
membrane potential (volts)
0.05
0
−0.05
−0.1
−0.15
−0.2
−0.25
0
0.1
0.2
0.3
time (sec)


1.0 nA
membrane potential (volts)
0.05
0
−0.05
−0.1
−0.15
−0.2
−0.25
0
0.1
0.2
time (sec)


model 1
model 2
model 3
model 4
mo
match (log units)
−2
−3
−4
−5
−6
−7
−8
−1
−2
−3
−4
−5
−3
−4
−5
−6
−7
−3
−4
−5
−1
−2
−3

parameter	description	mean	std. dev.	CV
1	$\overline{G}_{Na}$	3088.46	1885.78	0.61
2	$\overline{G}_{Na(pers)}$	108.27	38.77	0.36
3	$\tau(V)_{Na(pers)}$ (X)	1.30	0.44	0.34
4	$\overline{G}_{K(DR)}$	522.52	176.08	0.34
5	$\overline{G}_{K(A)}$	262.33	224.02	0.85
6	$\tau(V)_{K(A)}$ (X)	1.60	0.72	0.45
7	$\tau(V)_{K(A)}$ (Y)	1.72	1.03	0.60
8	$\overline{G}_{K(M)}$	96.31	35.35	0.37
9	$m_{\infty\ K(M)}$ offset (X)	0.0036	0.0038	1.04
10	$\tau(V)_{K(M)}$ (X)	0.67	0.25	0.37
11	$\overline{G}_{K(Ca,slow)}$	7.76	6.56	0.85
12	$\tau(Ca)_{K(Ca,slow)}$ (Z)	2.77	2.44	0.88
13	$\overline{G}_{K(Ca,fast)}$	19.40	7.84	0.40
14	$m_{\infty\ K(Ca,fast)}$ offset (Z)	1.53e-06	6.47e-07	0.42
15	$\tau(Ca)_{K(Ca,fast)}$ (Z)	2.73	2.48	0.91
16	$\overline{G}_{Ca}$	6.39	9.070	1.42
17	$\tau(V)_{Ca}$ (X)	1.04	1.11	1.07
18	$\tau(V)_{Ca}$ (Y)	0.67	0.48	0.72
19	Ca buffer B $(M/(A\ s))$	0.95	0.64	0.68
20	Ca buffer τ (sec)	1.99	0.88	0.44
21	resting potential (V)	-0.074	0.0020	0.027
22	soma RM $(\Omega\ M^2)$	0.54	0.21	0.38
23	dendritic RM $(\Omega\ M^2)$	0.52	0.11	0.22

Table 7: Variation in parameter values for successful parameter searches for model 5 (pyramidal cell model). $N = 10$ successful matches. Abbreviations are as in table 5.

parameter	description	range	original	SS	CG	ra
1	$\overline{G}_{Na}$	4.0	2800	1242.648	10582.60	71
2	$\overline{G}_{Na(pers)}$	4.0	65.00	153.735	124.612	6
3	$\tau(V)_{Na(pers)}$ (X)	4.0	1.00	1.652	1.735	
4	$\overline{G}_{K(DR)}$	4.0	270.00	1080.000	917.425	16
5	$\overline{G}_{K(A)}$	4.0	200.00	800.000	52.321	5
6	$\tau(V)_{K(A)}$ (X)	4.0	1.00	3.856	0.540	
7	$\tau(V)_{K(A)}$ (Y)	4.0	1.00	0.324	1.963	
8	$\overline{G}_{K(M)}$	10.0	100.00	22.305	59.084	10
9	$m_{\infty\ K(M)}$ offset (X)	0.010	0.000	-0.00367	-0.00684	
10	$\tau(V)_{K(M)}$ (X)	10.0	1.00	0.422	0.750	
11	$\overline{G}_{K(Ca,slow)}$	10.0	4.00	22.008	7.754	
12	$\tau(Ca)_{K(Ca,slow)}$ (Z)	10.0	1.00	4.488	3.312	
13	$\overline{G}_{K(Ca,fast)}$	10.0	20.00	6.547	47.232	7
14	$m_{\infty\ K(Ca,fast)}$ offset (Z)	3.0e-6	0.000	2.98e-06	2.25e-06	1.4
15	$\tau(Ca)_{K(Ca,fast)}$ (Z)	10.0	1.00	0.681	6.150	
16	$\overline{G}_{Ca}$	10.0	10.00	9.163	24.553	
17	$\tau(V)_{Ca}$ (X)	4.0	1.00	1.436	3.111	
18	$\tau(V)_{Ca}$ (Y)	4.0	1.00	0.480	1.066	
19	Ca buffer B $(M/(A\ s))$	4.0	1.83e+6	0.522	0.288	
20	Ca buffer τ (sec)	4.0	0.025	2.780	1.009	
21	resting potential (V)	0.005	-0.073	-0.0702	-0.0749	-0
22	soma RM $(\Omega\ M^2)$	2.0	0.50	0.676	0.462	
23	dendritic RM $(\Omega\ M^2)$	2.0	0.50	0.255	0.519	
	match			0.02379	0.02812	0.

Table 6: The target parameters for model 5 (pyramidal cell model) compared with the best parameters found by the SS, CG and random search methods. Abbreviations are as in table 5.

1	$\overline{G}_{Na}$	4.0	2800	3397.716	1403.483
2	$\overline{G}_{Na(pers)}$	4.0	65.00	128.717	160.279
3	$\tau(V)_{Na(pers)}$ (X)	4.0	1.00	1.091	1.285
4	$\overline{G}_{K(DR)}$	4.0	270.00	443.985	799.405
5	$\overline{G}_{K(A)}$	4.0	200.00	798.330	106.927
6	$\tau(V)_{K(A)}$ (X)	4.0	1.00	1.605	0.310
7	$\tau(V)_{K(A)}$ (Y)	4.0	1.00	2.064	1.575
8	$\overline{G}_{K(M)}$	10.0	100.00	112.174	143.803
9	$m_{\infty\ K(M)}$ offset (X)	0.010	0.000	0.009	0.006
10	$\tau(V)_{K(M)}$ (X)	10.0	1.00	0.414	0.575
11	$\overline{G}_{K(Ca,slow)}$	10.0	4.00	20.922	17.183
12	$\tau(Ca)_{K(Ca,slow)}$ (Z)	10.0	1.00	2.382	0.994
13	$\overline{G}_{K(Ca,fast)}$	10.0	20.00	20.886	16.425
14	$m_{\infty\ K(Ca,fast)}$ offset (Z)	3.0e-6	0.000	2.27e-06	7.74e-07
15	$\tau(Ca)_{K(Ca,fast)}$ (Z)	10.0	1.00	0.892	1.003
16	$\overline{G}_{Ca}$	10.0	10.00	6.630	1.862
17	$\tau(V)_{Ca}$ (X)	4.0	1.00	0.285	0.334
18	$\tau(V)_{Ca}$ (Y)	4.0	1.00	0.781	1.236
19	Ca buffer B $(M/(A\ s))$	4.0	1.83e+6	0.659	0.611
20	Ca buffer τ (sec)	4.0	0.025	2.130	3.028
21	resting potential (V)	0.005	-0.073	-0.074	-0.0763
22	soma RM $(\Omega\ M^2)$	2.0	0.50	0.411	0.707
23	dendritic RM $(\Omega\ M^2)$	2.0	0.50	0.440	0.540
	match			0.00615	0.009039

Table 5: The target parameters for model 5 (pyramidal cell model) compared with the best parameters found by the SA and GA methods. Maximal conductances ($\overline{G}$) are in S/m^2. Scaling factors for time constants τ are dimensionless. Offsets of m_∞ curves are in V (for X gates) or M (for Z gates). X gates are voltage-dependent activation gates, Y gates are voltage-dependent inactivation gates, and Z gates are calcium-dependent activation gates. Abbreviations for channels are as follows: Na: fast sodium; $Na(pers)$: persistent sodium; $K(DR)$: delayed-rectifier potassium; $K(A)$: inactivating potassium; $K(M)$: slow noninactivating potassium; $K(Ca, slow)$: slow Ca-dependent potassium; $K(Ca, fast)$: fast Ca-dependent potassium; Ca: voltage-dependent calcium.

5	dendrite 1 RA	6.0	3.01	3.06	4.14	10.9	
6	dendrite 1 CM	0.008	0.0101	0.016	0.0123	0.00634	
7	dendrite 2 RM	0.2	0.169	0.0977	0.225	0.146	
8	dendrite 2 RA	6.0	5.04	2.9	6.00	4.36	
9	dendrite 2 CM	0.008	0.0103	0.0185	0.0266	0.0233	0
10	dendrite 3 RM	2.0	3.56	13.8	0.25	2.12	
11	dendrite 3 RA	0.6	0.915	0.188	0.016	0.791	
12	dendrite 3 CM	0.008	0.0057	0.00323	0.004	0.00363	
13	dendrite 4 RM	2.0	2.54	4.73	2.47	0.729	
14	dendrite 4 RA	6.0	7.08	7.98	6.23	5.46	
15	dendrite 4 CM	0.08	0.0684	0.0654	0.0718	0.0757	
	match	0.0	2.14e-05	3.23e-05	7.38e-05	1.34e-04	1.

Table 3: The target parameters for model 4 (passive dendritic model with four branches) compared with the best parameters found by all search methods used. RM values are in units of $\Omega \cdot m^2$; RA values are in units of $\Omega \cdot m$, and CM values are in units of F/m^2. Note the relatively large spread of parameter values around the correct values.

parameter	description	target	mean	std. dev.	CV
1	soma RM	0.2	0.99	1.35	1.37
2	soma RA	6.0	3.66	4.17	1.14
3	soma CM	0.008	0.011	0.0072	0.66
4	dendrite 1 RM	2.0	1.01	0.70	0.69
5	dendrite 1 RA	6.0	2.91	1.89	0.65
6	dendrite 1 CM	0.008	0.014	0.011	0.74
7	dendrite 2 RM	0.2	0.146	0.057	0.39
8	dendrite 2 RA	6.0	4.41	1.72	0.39
9	dendrite 2 CM	0.008	0.0089	0.0068	0.76
10	dendrite 3 RM	2.0	5.25	7.72	1.47
11	dendrite 3 RA	0.6	1.02	0.99	0.97
12	dendrite 3 CM	0.008	0.0095	0.0101	1.06
13	dendrite 4 RM	2.0	2.81	1.25	0.45
14	dendrite 4 RA	6.0	7.27	1.60	0.22
15	dendrite 4 CM	0.08	0.068	0.013	0.18

Table 4: Variation in parameter values for successful parameter searches for model 4 (passive dendritic model with four branches). $N = 13$ successful matches.

3	$\overline{G}_{K(DR)}$	100.00	100.44	99.10	92.77	106.37	
4	$\tau(V)_{K(DR)}$ (X)	1.00	0.99	1.07	0.97	1.50	
5	$\overline{G}_{K(A)}$	50.00	49.79	50.00	53.11	26.38	
6	$\tau(V)_{K(A)}$ (X)	1.00	0.99	1.37	1.23	0.84	
7	$\overline{G}_{K(M)}$	100.00	100.30	95.41	93.85	99.35	
8	$\tau(V)_{K(M)}$ (X)	1.00	1.00	1.08	1.01	1.05	
	match	0.00	0.00	8.88e-04	2.46e-03	3.21e-03	6.5

Table 1: The target parameters for model 2 (simple spiking model) compared with the best parameters found by all search methods used. Maximal conductances are in S/m^2 and scaling factors for time constants are dimensionless. (X) refers to X gates which are voltage-dependent activation gates. $K(DR)$ is a delayed-rectifier potassium current; $K(A)$ is an inactivating potassium current; $K(M)$ is a slow non-inactivating potassium current. Note that the matches found by the SA and GA methods are very close to the correct parameter values.

parameter	description	target	mean	std. dev.	CV
1	$\overline{G}_{Na}$	800.00	782.41	200.14	0.26
2	$\tau(V)_{Na}$ (X)	1.00	0.83	0.35	0.42
3	$\overline{G}_{K(DR)}$	100.00	96.79	13.89	0.14
4	$\tau(V)_{K(DR)}$ (X)	1.00	1.05	0.12	0.11
5	$\overline{G}_{K(A)}$	50.00	45.91	10.18	0.22
6	$\tau(V)_{K(A)}$ (X)	1.00	1.09	0.47	0.43
7	$\overline{G}_{K(M)}$	100.00	98.10	6.97	0.071
8	$\tau(V)_{K(M)}$ (X)	1.00	1.02	0.080	0.079

Table 2: Variation in parameter values for successful parameter searches for model 2 (simple spiking model). $N = 13$ successful matches.

some cases adapted to use calcium concentration rather than membrane potential as the independent variable for the activation and inactivation curves. The equation we used for calcium buffering in the pyramidal cell model is the following:

$$d[Ca]/dt = B \cdot I_{Ca} - [Ca]/\tau$$

where $[Ca]$ is the calcium ion concentration, τ is the buffer time constant, and B is a scaling factor which converts calcium current influx into changes in calcium concentration. B and τ were both parameters of the model; their initial values and ranges are in table 5.

not precisely reproduce every aspect of the interspike interval waveforms the match between the spike times of the model compared to the data is extremely good.

Fig. 7: Comparison of the performance of the parameter-searching methods on a model of a layer 2 pyramidal cell from piriform cortex with 23 parameters (model 5) as a function of simulation number. All traces are the averages of ten runs of 100,000 simulations each.

Fig. 8: Results of genetic-algorithm parameter searches on the pyramidal cell model with and without the addition of a hypothetical fast AHP channel. The addition of the fast AHP channel allowed the parameter search to find significantly better matches.

Fig. 9: Outputs of a layer 2 pyramidal cell from piriform cortex at six different levels of current injection at the soma compared with the outputs of a genetic-algorithm parameter search. This model did not contain a fast AHP channel. Note that the model output is not as close to the real data as is figure 6, which is derived from a model containing a fast AHP channel.

the SA results for models 1 and 2 are due to the presence of several match values of zero in the final results; however, a one-tailed Mann-Whitney U test confirmed that the SA match values are significantly smaller than those of the other methods in both cases ($p < 0.01$). Note the different y-axis scales for the different graphs.

Fig. 2: Target outputs of the simplified one-compartment spiking model. The graphs show the membrane potential changes in response to current steps of various amplitudes, from 0.1 nA to 1.0 nA. The lower traces represent the target data, while the upper traces represent the results of a genetic-algorithm parameter search using 8 parameters (model 2). The lower traces have been offset by -150 mV for clarity. As can be seen, the parameter search found an almost perfect match to the target data set.

Fig. 3: Left graph: comparison of the performance of the parameter-searching methods on the simple spiking model with four parameters (model 1) as a function of simulation number. The simulated annealing method dramatically outperformed all other methods in both the ultimate quality of the match achieved and the speed at which good matches were obtained. The other methods were only marginally better than random search. Right graph: the same comparison for the eight-parameter version of the simple spiking model (model 2). The simulated annealing method is again the superior method, but the degree of superiority is less marked that for model 1. All of the other methods give better results than pure random search. All traces are the averages of ten runs of 10,000 simulations each.

Fig. 4: Target outputs of the passive dendritic model with 15 parameters (model 4) compared to the results of a simulated-annealing parameter search. The target data is indicated by the broken line and the results of the parameter search are indicated by the solid line. The target data is offset by -2 mV for display purposes; otherwise the traces would overlap entirely.

Fig. 5: Left graph: comparison of the performance of the parameter-searching methods on the linear dendritic model with three parameters (model 3) as a function of simulation number. Once again, the simulated annealing method is dramatically superior to the other methods on average. Note the relatively good performance of the CG method. Right graph: comparison of the performance of the parameter-searching methods on the linear dendritic model with fifteen parameters (model 4) as a function of simulation number. The simulated annealing method is still superior to the other methods on average, but the difference is considerably smaller than for model 3. All traces are the averages of ten runs of 10,000 simulations each.

McCormick, D. A., Connors, B. W., Lighthall, J. W., and Prince, D. A. (1985). Comparative electrophysiology of pyramidal and sparsely spiny stellate neurons of the neocortex. *Journal of Neurophysiology*, 54(4):782–806.

McCormick, D. A. and Huguenard, J. R. (1992). A model of the electrophysiological properties of thalamocortical relay neurons. *Journal of Neurophysiology*, 68(4):1384–1400.

Mitchell, M. (1996). *An introduction to genetic algorithms.* The MIT Press, Cambridge.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical Recipes in C, 2nd. Ed.* Cambridge University Press, New York.

Rall, W. (1995). *The Theoretical Foundation of Dendritic Function.* MIT Press, Cambridge, Mass.

Rall, W., Burke, R. E., Holmes, W. R., Jack, J. J. B., Redman, S. J., and Segev, I. (1992). Matching dendritic neuron models to experimental data. *Physiological Reviews*, 72(4):S159–S186.

Rinzel, J. (1985). Excitation dynamics - insights from simplified membrane models. *Federation Proceedings*, 44(15):2944–2946.

Staley, K. J., Otis, T. S., and Mody, I. (1992). Membrane-properties of dentate gyrus granule cells - A comparison of sharp microelectrode and whole-cell recordings. *J. Neurophysiol.*, 67(5):1346–1358.

Tawfik, B. and Durand, D. M. (1994). Nonlinear parameter-estimation by linear association - application to a 5-parameter passive neuron model. *IEEE Transactions on Biomedical Engineering*, 41(5):461–469.

Traub, R. D., Wong, R. K. S., Miles, R., and Michelson, H. (1991). A model of a CA3 hippocampal pyramidal neuron incorporating voltage-clamp data on intrinsic conductances. *Journal of Neurophysiology*, 66(2):635–650.

Vanier, M. and Bower, J. (1996). A comparison of automated parameter-searching methods for neural models. In Bower, J. M., editor, *Proceedings of the 1995 Computational Neuroscience Conference (CNS*95), Monterey (CA).* Academic Press.

White, J. A., Manis, P. B., and Young, E. D. (1992). The parameter-identification problem for the somatic shunt model. *Biological Cybernetics*, 66(4):307–318.

pages 911–917. Plenum Publishing, New York.

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning.* Addison-Wesley, Reading, Mass.

Halliwell, J. V. and Scholfield, C. N. (1984). Somatically recorded Ca-currents in guinea-pig hippocampal and olfactory cortex neurons are resistant to adenosine action. *Neuroscience Letters*, 50(1-3):13–18.

Hille, B. (1992). *Ionic Channels of Excitable Membranes, 2nd. Ed.* Sinauer Associates Inc., Sunderland, Massachusetts.

Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems, 2nd. ed.* The MIT Press, Cambridge.

Holmes, W. R. and Rall, W. (1992a). Electrotonic length estimates in neurons with dendritic tapering or somatic shunt. *Journal of Neurophysiology*, 68(4):1421–1437.

Holmes, W. R. and Rall, W. (1992b). Estimating the electrotonic structure of neurons with compartmental models. *Journal of Neurophysiology*, 68(4):1438–1452.

Jack, J. J. B., Noble, D., and Tsien, R. W. (1983). *Electric Current Flow in Excitable Cells.* Clarendon Press, Oxford.

Keeping, E. S. (1995). *Introduction to Statistical Inference.* Dover Publications, Inc., New York.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Koch, C. and Segev, I. (1989). *Methods in Neuronal Modeling.* MIT Press, Cambridge, Massachusetts.

Koch, C. and Segev, I. (1997). *Methods in Neuronal Modeling, 2nd. Ed.* MIT Press, Cambridge, Massachusetts.

Koza, J. R. (1992). *Genetic programming: on the programming of computers by means of natural selection.* The MIT Press, Cambridge.

Lemasson, G., Marder, E., and Abbott, L. F. (1993). Activity-dependent regulation of conductances in model neurons. *Science*, 259(5103):1915–1917.

Banks, M. I., Haberly, L. B., and Jackson, M. B. (1996). Layer-specific properties of the transient K-current (I-A) in piriform cortex. *Journal of Neuroscience*, 16(12):3862–3876.

Bhalla, U. S. and Bower, J. M. (1993). Exploring parameter space in detailed single neuron models - simulations of the mitral and granule cells of the olfactory-bulb. *Journal of Neurophysiology*, 69(6):1948–1965.

Bower, J. M. and Beeman, D. (1995). *The Book of Genesis*. Springer Verlag, New York.

Bush, P. C. and Sejnowski, T. J. (1993). Reduced compartmental models of neocortical pyramidal cells. *Journal of Neuroscience Methods*, 46:159–166.

Buzsaki, G. and Kandel, A. (1998). Somadendritic backpropagation of action potentials in cortical pyramidal cells of the awake rat. *Journal of Neurophysiology*, 79(3):1587–1591.

Constanti, A., Galvan, M., Franz, P., and Sim, J. A. (1985). Calcium-dependent inward currents in voltage-clamped guinea-pig olfactory cortex neurons. *Pflugers Archiv European Journal of Physiology*, 404(3):259–265.

DeSchutter, E. and Bower, J. M. (1994a). An active membrane model of the cerebellar Purkinje cell I. simulation of current clamps in slice. *Journal of Neurophysiology*, 71:375–400.

DeSchutter, E. and Bower, J. M. (1994b). An active membrane model of the cerebellar Purkinje cell II. simulation of synaptic responses. *Journal of Neurophysiology*, 71:401–419.

Eichler-West, R. and Wilcox, G. (1997). Robust parameter selection for compartmental models of neurons using evolutionary algorithms. In Bower, J. M., editor, *Computational Neuroscience: Trends in Research 1997*. Plenum Publishing, New York. In press.

Foster, W. R., Ungar, L. H., and Schwaber, J. S. (1993). Significance of conductances in Hodgkin-Huxley models. *Journal of Neurophysiology*, 70(6):2502–2518.

French, C. R., Sah, P., Buckett, K. J., and Gage, P. W. (1990). A voltage-dependent persistent sodium current in mammalian hippocampal-neurons. *Journal of General Physiology*, 95(6):1139–1157.

criteria for comparing the quality of different models which are intended to match the same data set. We have recently begun exploring this issue, using Bayesian methods to compare neural models quantitatively (Baldi et al., 1998). As neural models become more complex the subject of parameter-searching in these models will continue to develop and will be critical to progress in computational neuroscience.

(tens to hundreds of thousands for the models we investigated), the time required to simulate a single iteration of a model must be relatively small. For instance, a model which requires one hour or more to go through one iteration (*e.g.* (DeSchutter and Bower, 1994a; DeSchutter and Bower, 1994b)) will take over a year of workstation time to run through 10,000 simulations. Obviously, unless the simulation time can be reduced significantly by running on faster computers (*e.g.* parallel supercomputers (Goddard and Hood, 1997)), such a model is currently not a good candidate for automated parameter searches. Under these conditions parameter searches can sometimes still be run on simplified versions of a large model if the results can then be applied as a constraint to the larger model.

At the present time it does not appear that the parameter search methods we describe will be useful on models with huge numbers (*e.g.* hundreds) of parameters. We note that moving from a model with eight parameters to one with 23 parameters required a scale-up of 10 in the number of iterations per parameter search needed to get good results. It is likely that models with very large numbers of parameters (> 100) can easily outrun the capabilities of the current generation of workstations. This is a key argument for developing parameter-search methods to work on parallel supercomputers, as we have begun doing (Goddard and Hood, 1997). In addition, it is desirable for modelers to investigate ways of reducing the size of the parameter spaces of their models, for instance by forcing some parameters in different compartments to have the same values (*e.g.* (DeSchutter and Bower, 1994a)).

Another potential difficulty with parameter searches concerns overfitting of the data. When a model is carefully tuned to match a specific data set, it may not generalize well to similar data sets not used in the tuning. For instance, if a model is tuned to match the spike times very precisely at a given value of current injection it may fail to match the spikes times at other current levels. For this reason, our own studies employ a wide range of inputs to provide an appropriate breadth of constraints on model behaviors. Such models are far more likely to be underfit than overfit. If overfitting is suspected, it is a simple matter to parameterize one's model on a subset of the data set (the training set) and see if once the model is parameterized it also matches the rest of the data (the test set).

It is also critical not to place too much weight on the results of a parameter search in a weakly-constrained or highly simplified model. In our pyramidal cell model, many of the kinetic parameters were only approximately known, those relating to calcium buffering were only guesses, and we added a hypothetical fast AHP current. However, it is entirely possible that a combination of different calcium buffering schemes, dendritic active currents, or other channels we did not include could have given equally good or better results. For this reason, our modeling results allow us to speculate about the possible significance of the hypothetical fast AHP channel,

of robustness, however, this will dramatically increase the computational cost of the parameter search. A better alternative is to test the robustness of the best parameter sets from a search and to reject those which are not sufficiently robust.

Another kind of robustness is robustness to small changes of the inputs. For example, one can make multiple measurements of the output of the cells to quantify the variability of the data source, introduce variability into the model in any of a number of different ways, and use Bayesian methods to determine the probability of the model given the data (Baldi et al., 1998). This allows modelers to test the robustness of their models to small perturbations of the inputs in a rigorous way.

4.6 Recommendations for effective parameter searching

While we have compared the general usefulness of different parameter-searching algorithms and discussed matching functions, the process of parameter searching and the interpretation of the results can be severely hindered or improved by other factors which we discuss here.

An important issue in setting up a parameter search concerns the process of setting the parameters of the search algorithm itself. The ranges of the parameters being searched are important parameters for search methods; ideally they should be set to physiological ranges but in many cases not enough is known to constrain these ranges. If the true parameters lie outside the initially established range, then the searching algorithm will not be able to find them. Additionally, all search algorithms have parameters of their own, such as the annealing rate for the SA method and the population size, mutation rate and recombination rate for the GA method. Unfortunately, currently there is little analytical work which can guide the choice of these parameters, so they tend to be selected by experience and *ad hoc* rules of thumb (Goldberg, 1989; Mitchell, 1996).

It is also necessary that the components of the model be carefully selected. A great deal of wasted time can be avoided, for example, if the initial state of the model at least qualitatively matches the gross features of the target data set. Parameter searches do not remove the need for good judgment on the part of the modeler, and cannot make a bad model good. As a corollary, however, we have often found that parameter searches are often more useful when they fail than when they are successful. A failure in a well-executed parameter search suggests that a model may be incapable of reproducing the specified behavior (at least within the allowed parameter ranges). This failure naturally leads to further experimentation with either the model or the real system. For example, one can explore whether adding a new ionic channel to the model would make the model match the experimental data much more closely (*e.g.* figures 6 and 9). If so, this can suggest new ideas for experiments.

search methods by calculating statistics on a parameter-by-parameter basis for the best matches found by all the methods. The best parameter sets found by the different methods in models 1 and 3 (with 4 and 3 parameters, respectively) matched the correct parameter sets very closely, with little variation in the parameter values. This suggests that the parameter spaces for these models are dominated by a single deep minimum which was reliably found by many of the searches (in particular the SA and GA searches). Parameter searches on model 2, with 8 parameters, also gave rise to parameter values that clustered around the correct values, albeit with considerably greater variance (table 2). Models 4 and 5 (with 15 and 23 parameters, respectively) showed a very high degree of variation in the parameter values obtained between different successful matches (tables 4 and 7). This suggests that parameter spaces with larger numbers of parameters show more degeneracy in their match values *i.e.* there is more than one equally good location in parameter space which can be discovered by the search algorithms. Not surprisingly, some parameters are constrained more than others. For model 2, the parameters controlling the maximal conductance and kinetics of the slow potassium M-current displayed a considerably smaller coefficient of variation (CV) than the other parameters, which suggests that these parameters need to be located in a narrow range for good matches to be achieved. Since in this model the M-current is the major controller of the spike adaptation behavior of the cell, these results are not surprising. In model 5 the variability of the somatic membrane potential was comparatively small, which indicates that varying this parameter would adversely affect the match value. Shifting the membrane potential can be expected to affect the excitability of the cell due to the increased (or decreased) activation of voltage-dependent ionic channels. Model 4 shows a high degree of variability for all parameters; this suggests that multiple good solutions exist for this model, which in turn indicates the need to use more data to parametrize the model if unique matches are desired.

4.5 Robustness of the parameter sets

One question of interest to modelers using parameter searching techniques is the degree to which parameter sets are robust to small perturbations in the parameter values. For instance, if parameter values need to be specified with great precision in order for a good match to be achieved, the parameter set is not robust and therefore is unlikely to be physiologically relevant. In practice, estimating robustness of parameter sets is difficult because one would ideally like to vary all possible combinations of parameters. However, some insight into robustness can be achieved by varying each parameter individually (or pairs of parameters (Bhalla and Bower, 1993; Foster et al., 1993)) around its initial value while keeping the other parameters fixed and measuring the change in the match value. Robustness estimated in this

we found that this approach did not sufficiently penalize models whose behaviors were clearly qualitatively incorrect. For example, some models with particular parameter settings generated small bursts of spikes (*e.g.* pairs of spikes separated by a small ISI) separated by a much larger interburst interval, in contrast to the more uniform spiking pattern we were trying to replicate. While this behavior would eventually have been selected against even using an RMS error function, we found it more efficient to put a factor into our matching function to directly penalize bursting. This allowed us to narrow down the effective search space enormously and thus sped up the searches. To be most effective, considerable thought must be applied to the selection of matching functions to achieve optimal performance. Often, initial experience with a matching function reveals that some important aspect of the data was overlooked. Therefore it is often useful to examine how well the matching function distinguishes good from bad models before committing large amounts of computer time to an extensive parameter search.

The choice of matching function can also contribute significantly to the computational cost of the parameter search, because a match must be computed for each simulation run. For this reason we chose to use spike times exclusively as the criteria for matching models 1, 2, and 5. If computational costs were not an issue, it would have been preferable to use a combination of spike timings and interspike interval (ISI) waveforms to match these models. However, since most models generate relatively few spikes, a spike timing based match function is much more efficient to calculate than a match function which also includes ISI waveforms, because for the latter the membrane potential value at each time point must be used in the computation. For some models, however, the duration of the individual simulations will still be much larger than the time needed to compute a match even if ISI waveforms are used in the match function. In such cases, one should use ISI waveforms as well as spike timings in the match function. For the passive models we had no choice but to use membrane potential waveforms, since the models do not generate spikes.

4.4 Variation in the parameter sets

Depending on the goals of the modeler, he may or may not be interested in the question of whether different runs of a parameter search result in the same set of parameter values. On one extreme, the model may be used as a phenomenological model which is a considerable abstraction of the real system, and the modeler may only wish to have a set of parameters that produces behavior similar to that seen in the real system. In this case the uniqueness of the parameter set is of little interest. At the other extreme, a modeler may aspire to build a very detailed model whose parameters (some measured, some estimated) accurately represent the corresponding real parameters of the system. In this case the modeler is very interested in whether

method proved highly effective was the 3-parameter passive model (model 3) which, due to its linearity and the small number of parameters, is more likely to have a smoothly varying parameter space.

Within the class of stochastic models, our results clearly demonstrate differences between genetic algorithms (GA), stochastic search (SS) and simulated annealing (SA). The SS method was markedly inferior overall to both the SA and the GA method, which is almost certainly due to the extremely unsophisticated search strategy this method employs. In general, we obtained better results with the SA method than the GA method. The SA method we used also possesses most of the advantages of gradient-descent methods without their disadvantages. At high temperatures the method can explore large expanses of parameter space, and as the temperature is lowered, the search narrows in on specific regions with better match values. The temperature-dependent noise in the method prevents the search from getting stuck in local minima. When the temperature approaches zero the SA method converges to the closest local minimum as does a gradient-descent search. (In contrast, the GA method never converges to a local minimum of the match function). However, our SA method can be expected to give poorer results with larger models, as this method can only keep a record of a small number of points at a time (the points on the simplex, which number one more than the number of dimensions). For a search with a large enough number of parameters the GA method, which can simultaneously search many points in parameter space, may well overtake the SA method in performance. For our models the GA method was as effective as the SA method for the largest model (model 5). It is likely that for an even larger model GAs would have been superior. Work on GAs (Goldberg, 1989; Holland, 1992; Koza, 1992; Mitchell, 1996) has suggested that this method is often optimal for problems when a large number of largely independent factors must be combined to create the best results. In contrast, in compartmental neural models optimal parameter values are typically highly dependent on each other, especially within a compartment, so that changing the value of one parameter will require changes in the values of all other parameters as well to optimize the match. This would seem to indicate that GAs are poorly suited for compartmental models. However, while GAs are less effective for problems containing large numbers of strongly interacting parameters, they are capable of independently exploring large segments of parameter space simultaneously and in parallel. This may accounts for their greater relative usefulness with larger models. The intrinsic parallelism of GAs also makes them very well suited for implementation on parallel computers (Eichler-West and Wilcox, 1997; Goddard and Hood, 1997).

(table 2). In contrast, models 4 and 5 showed a very wide range of parameter values for successful matches (tables 4 and 7).

4 Discussion

4.1 Method comparisons

There are two major conclusions that can be drawn from the results presented. First, parameter-search methods can be extremely effective in finding very good matches between single-neuron models and a target data set in a wide variety of cases. Second, not all parameter-search methods perform equally well, and the relative performance of the different methods depends on the model being optimized. In general, the simulated annealing method and the genetic algorithm method were superior to the other methods we investigated (stochastic search and conjugate gradient) on all of our models (except for model 3, the linear passive dendrite model, for which the conjugate gradient method was as good as the genetic algorithm method). The reader should be aware, however, that all of our parameter-searching methods have parameters of their own which we have determined by experimentation (see below). Therefore, it is possible that different choices of these meta-parameters could have resulted in somewhat different results. In addition, variations of each of these methods exist, especially for genetic algorithms (Mitchell, 1996). More sophisticated algorithms in each class may very well have improved the performance of our searches. For this reason the reader should consider the data presented as a general guide providing an upper bound on the performance of these methods on the models examined.

4.2 Differences between the search algorithms

While it is useful to know which parameter searching algorithms are most appropriate for single-neuron compartmental models of different complexity, it would also be useful to know why one algorithm was more effective than another for a given model. Unfortunately, understanding these relationships rigorously is very difficult as it involves an interaction between the searching algorithm and the structure of the model's parameter space. However, based on previous analysis of this type (Bhalla and Bower, 1993; Foster et al., 1993), it is possible to speculate on several possibilities for the results we obtained.

In general, it is likely that stochastic methods rather than those based on gradient-descent will be more successful with the types of models studied here. The reason for this is that simple gradient-descent methods such as conjugate gradients are greedy, *i.e.* they move to the nearest local minimum in parameter space and stay there.

criteria. However, figure 6 shows that the methods can achieve extremely good matches to the data over a wide range of input currents, at least with respect to the spike timings. The major difference between the model and the data is in the spike pre- and afterpotential, suggesting that other conductances or features need to be added to the model if greater accuracy is needed or if the spike afterpotential is of interest to the modeler. One interesting possibility would be to add a separate spike-initiation zone to the model (Mainen et al., 1995). An interesting feature of this model is that we added a fast calcium-dependent potassium channel to the model, even though there is no experimental evidence for such a channel, because we found that adding this channel substantially improved the matches we obtained (see below and Discussion). Figure 7 shows a graph of the results of the different methods as a function of simulation number, while figure 1 shows the statistical distribution of the best match results. Note that for this model the genetic algorithm method was almost exactly as effective as the simulated annealing method. The CG and SS method are roughly equivalent, while (as expected) the random search method is the least effective.

As mentioned above, we added a hypothetical fast AHP channel to give better matches between the data and the model. As a control we also ran a series of parameter searches without the fast AHP channel using the genetic algorithm method (the SA method could also have been used but was not). Figure 8 compares the results of the GA simulations with and without the fast AHP channel. As can be seen, the addition of the fast AHP channel improved the match value considerably. The best model we obtained using a parameter search without a fast AHP channel is shown in figure 9. Although this model matches the data fairly well, it is markedly inferior to the model in figure 6, which did include the fast AHP channel. The difference between the results of the GA searches for the model without the fast AHP channel and the model with the fast AHP channel is statistically significant ($p < 0.05$, two-tailed t-test).

3.4 Variation in parameter sets obtained

We examined the variation in the parameter values for the matches obtained by the different search methods. Since it is only reasonable to compare parameter sets from successful parameter searches, for each model we took the best results from each round of all methods and selected the parameter sets whose matches were within 10% of the best match value, which we designate as successful matches. For all models except for model 3, all successful matches using our criterion were derived either from the SA or GA method; for model 3, the CG method was also frequently successful. Parameter variation was quantified by calculating the coefficient of variation (CV, the standard deviation divided by the mean) of the parameter values found by the

those of the other methods ($p < 0.01$).

3.2 Passive cable models

See figure 4 for typical outputs of model 4, with 15 parameters. In this figure the target data is plotted using dashed lines while the model found by the parameter search is plotted using solid lines. The target data has been offset by -2 mV for clarity; if we had not done this the models would overlap completely. The outputs from model 3 were equally impressive. In table 3 we show the parameters of the target model versus those of the model found by the parameter search for model 4. It is extremely interesting that, unlike the case with the simple spiking model, the parameter values found for this model were in many cases quite different from the correct parameters even for models which match the data very closely. This indicates that in this model there can be many optimal models based on our criteria. Another interesting point is that the capacitance values in the best parameter set (found using the SA method) were generally quite close to the correct values, whereas the resistance parameters often varied considerably from the corresponding correct values. This is not surprising since the time course of the voltage waveforms is critically dependent on the capacitance values, whereas the various resistance values can compensate for each other to some extent (*e.g.* if one dendrite has a higher-than-average input resistance, the other can have a lower-than-average one).

Figure 5 demonstrates how well the different search methods work as a function of simulation number for the passive models, while figure 1 shows the final results of the parameter search methods along with the range of the results. Simulated annealing still outperformed the other methods, with genetic algorithms second as before. However, for model 3 the conjugate gradient (CG) method significantly outperformed the stochastic search (SS) method ($p < 0.05$) and performed as well as the genetic algorithm method. This presumably reflects the fact that for passive models with small numbers of parameters the parameter space is relatively smooth, so a gradient-descent method like the CG method can be reasonably effective. Interestingly, when we look at model 4, which is more complex, the CG method gives slightly worse results than the SS method (although this is not statistically significant). This may reflect the larger number of models with equivalent match values for this model i.e. the larger number of local minima for the CG method (but not the SS method) to get trapped in.

3.3 Pyramidal cell model

Figure 6 shows the outputs of a parameter searching method for the pyramidal cell model. This model was discovered by the SA method after about 48,000 simulations.

1-4 10,000 simulations were performed for each run of a parameter search and 10 parameter searches were run for each model and each method in order to generate statistics. Thus we performed 4 (*models*) $*$ 5 (*methods*) $*$ 10 (*repeats*) $*$ 10000 (*simulations per search*) $=$ 2,000,000 simulations for these four models. For model 5 we found that we could not obtain reasonable results with 10,000 simulations, so we used 100,000 simulations per run instead. For this model we thus performed $5 * 10 * 100000$ or 5 million simulations.

3 Results

3.1 Simple spiking models

A typical result of a parameter search on model 2 (eight parameters) is shown in figure 2. This model was discovered by the genetic algorithm method. In the figure the target data is the bottom trace on each graph and has been offset to facilitate visual comparison. Note that the membrane potential trace of the model found by the search is nearly identical to that of the target model. The parameters of the model are listed in table 1 along with the best parameter sets found by the other search algorithms. The best parameters for the SA and GA methods are very close to the (known) parameters of the target model, while those of the other methods in some cases diverge considerably from the correct values. Even though the simulated annealing method gave better matches than the genetic algorithm method for model 2, the matches from both methods are sufficiently good that they would be difficult to distinguish by eye.

Figures 1 and 3 shows how the searching methods compare for models 1 and 2. Each round consisted of 10,000 searches and each line in figure 3 is the average of ten rounds. For both models the simulated annealing method was superior to the other methods both in terms of how good a match was finally achieved and how quickly this match was achieved. Random search was predictably the poorest method with the other three methods ranking in between. The best (*i.e.* lowest) match obtained for the SA method for both models was 0.0 (a perfect match). This means that each spike in the model found by the search matched the time of the target model within the resolution of the simulation output, which was 0.1 *msec*. Searches on the eight-parameter model generally resulted in higher match values than those for the four-parameter model. This reflects the larger size of the parameter space and thus the greater difficulty of the search task. Interestingly, the GA method gave similar match values for both models; this may reflect the fact that GA methods are less sensitive to the dimensionality of the problem (see Discussion).

The statistical distribution of the best match results is displayed graphically in figure 1. For model 1, the difference between the non-SA methods is not significant;

allow the modeler to find locally optimal points in small regions of parameter space. Unlike the SA algorithm, however, the SS algorithm does not do a systematic search for local minima.

2.5.5 Random search

As a control on the performance of the other parameter-search methods, we used a simple random search method as follows. For each simulation a new parameter set was chosen at random from the parameter space being searched (which always had a finite volume). The best parameter set was always saved, and the performance of the random search method equaled the best match obtained after a given number of simulations.

2.6 Statistical analyses

For each pair of parameter search method and model, we performed 10 separate parameter searches and recorded the best match values for each search. To compare the different methods, we computed the mean and 95% confidence intervals of the log transform of the best match values. The effect of the log transformation was to change the range of the matches from $[0, \infty]$ to $[-\infty, \infty]$. This was necessary since otherwise the confidence intervals would have in some cases included negative match values, which are inadmissible. For the time-course plots we plotted the arithmetic mean of the match values at each time point without log transformation. We separately calculated the geometric mean of the match values, which are equivalent to the exponential of the arithmetic mean of the log-transformed data. The results were qualitatively identical, with one exception: for the CG method in model 3 the geometric mean is lower than that of the GA method, while the arithmetic mean is higher; the difference is not significant in either case.

In two cases (the simulated annealing results of models 1 and 2) matches of zero were sometimes obtained, which made it impossible to log transform the matches. In these cases we separately calculated the significance of the difference between the simulated annealing results and those of the other methods using the nonparametric Mann-Whitney one-tailed U test (Keeping, 1995). For plotting purposes we replaced the zero match values by 10^{-8}, which gave relatively large confidence limits.

2.7 Resource requirements

For each of the parameter search methods, the computational overhead (in both space and time) of the search algorithm was negligible in comparison to that of the neural simulation itself. Therefore, we can compare the performance of the different methods solely by how good a match is obtained with a fixed number of simulation

nonzero probability which decreases to zero as the temperature falls. This allows the algorithm to escape from local minima. The annealing schedule was a simple proportional decrease in temperature every N simulation iterations, where N varied from 5 for the simple spiking models to 300 for the pyramidal cell model. We found the slower annealing schedule necessary to give good matches for the pyramidal cell model. With large amounts of noise, the simplex essentially performs a random walk in the parameter space, and as the noise level decreases the simplex hones in on regions of the parameter space with better match values. As noise is reduced to zero this method finds a local minimum in the parameter space, a useful property not shared by the other methods (nor by all simulated annealing algorithms).

The only difference between our procedure and that of (Press et al., 1992) was that in the latter the parameter space was infinite; we changed this by enforcing wraparound boundary conditions, so that if the algorithm tried to use a parameter value that was beyond the assigned limits of the parameter space, the value was adjusted to come in from the opposite side of the parameter space by the amount of the overshoot. This was necessary since otherwise the algorithm would often set parameters to extremely small or large (*i.e.* unphysiological) values in order to achieve very small improvements in the match. This also allowed us to enforce our constraint of keeping parameter values within specified limits without simply truncating parameter values at the limits.

If the SA algorithm converged before the parameter search run had used up all of its allocated simulations (which was always the case given the annealing schedules we used) the best results of the search were saved and we restarted the search from scratch. When plotting the results, we always give the best results of the run achieved up to that point, even when there was more than one search in the run. This was also true for the conjugate-gradient (CG) search method (described below).

2.5.4 Stochastic search

The stochastic-search (SS) method (Foster et al., 1993) works as follows. A random starting point in parameter space is chosen. New points are selected from a multidimensional Gaussian distribution centered on the starting point with a given (initially large) variance. These new points are evaluated, and if one is found that is a better match than the previous best match, the Gaussian distribution is moved so that it is centered on the new (best) point. In addition, the variance of the distribution decreases linearly each simulation until it reaches some minimum value, at which time it is increased to the original value. The variance value represents a proportion of the range of the parameter. We used an initial variance of 0.5, a final variance of 0.05, and a variance-contraction of 0.95 per simulation for all models except the pyramidal cell model, where we used a variance-contraction of 0.98 in

sets at random, choosing a breakpoint within the bit string and exchanging the bit strings above the breakpoint (single-point recombination). We crossed over $\frac{2}{3}$ of the parameter strings per generation. Finally, each parameter set was subjected to mutation with a low probability per bit. We used a mutation probability of 0.02 per bit for models 1-4 and 0.01 for model 5. In this way, highly fit parameter sets are selected for and less fit sets are eliminated from the population over a series of generations. Furthermore, the processes of crossing-over and recombination can generate new parameter combinations whose fitness is greater than that of its progenitors.

For models 1-4 we used a population size of 100 and ran the parameter search for 100 generations, giving a total of 10,000 simulations per round. For model 5 we used a population size of 300 and ran the search for 334 generations giving a total of about 100,000 simulations. These population sizes are quite small for GAs (Goldberg, 1989; Mitchell, 1996), but since the total number of simulations was limited (intentionally) if a larger population size had been used the simulation would have had to have been run for fewer generations. We found that these population sizes gave reasonable results but we did not investigate the population-size dependence of the GA systematically.

Our GA method had two nonstandard features: first, the best parameter set was always preserved unaltered. This was to prevent genetic drift from eliminating good parameter sets and was necessary since the small population sizes used meant that random fluctuations could have an excessively large effect on which parameter sets were propagated into the next generation. Second, if the method went a large number of generations without improving on the best parameter set, the entire parameter table (aside from the best parameter set) was reseeded by choosing random values for all parameter sets. This provided an additional way to keep the method from getting stuck in a suboptimal region of parameter space.

2.5.3 Simulated annealing

We used a continuous version of simulated annealing (SA) (Kirkpatrick et al., 1983) adapted from Press et. al. (Press et al., 1992). This algorithm constructs an N-dimensional figure known as a simplex out of $N+1$ points in the parameter space. In the deterministic form of the algorithm, the simplex is moved through the parameter space based on the calculated match values at the vertices of the simplex. Roughly speaking, successive points in the parameter space are evaluated and accepted if their matches are better than that of the worst point on the simplex (in which case the worst point is discarded and the new point joins the simplex) or rejected. This process continues until a local minimum is reached. In the noisy version, the match values for each point are modified with noise which varies linearly with a temperature parameter which is set by an annealing schedule. The temperature

parameter space and the direction of steepest descent with respect to this function are computed. A line minimization in the direction of steepest descent is done to find the point on the line with the best match value. Once this is done the procedure is iterated (with a slight variation) until the method converges to a local minimum of the matching function. The variation is that successive line minimizations are done in directions which are conjugate to one another *i.e.* they are chosen so that successive minimizations are as nearly independent of each other as possible. In theory this will guarantee that the parameter search will find a local minimum of the objective function. However, this method has certain difficulties when used with spiking neural models. First, we found that when using the spike timing-based match function there were large regions of parameter space which were completely flat *i.e.* where no gradient existed. This generally occurred where no spikes were produced by models whose parameters were in that region. Since no gradients could be calculated in these regions, the CG method could not be used directly. We dealt with this by running 100 random simulations before beginning the CG search and using the best match as a starting point. This usually resulted in a point where a gradient could be calculated. In addition, the process of numerically estimating gradients is very costly, requiring numerous simulation runs per gradient calculation. Minimization methods which find local minima without requiring explicit gradient calculations exist (*e.g.* Powell's method; see (Press et al., 1992)); however these methods require a large number of line minimization steps and it is not clear whether they would be any more efficient than the CG method.

2.5.2 Genetic algorithm

The genetic algorithm (GA) method (Goldberg, 1989; Holland, 1992; Koza, 1992; Mitchell, 1996), treats each parameter set as an individual in a large breeding population. A new generation of the population is derived from the preceding generation by reproduction, crossing-over and mutation. This was accomplished by discretizing the parameter values into bit strings and crossing-over and mutating the different bit strings. Parameters were represented as either 8-bit or 32-bit strings (8 bits for models 1 to 4; 32 bits for model 5); we found that the bit resolution had a minimal effect on the results. We took a population of parameter sets and evaluated the fitness of each one by computing the match as described above. Since the fitness function must increase for better matches, we calculated it by taking either the inverse of the match function (for model 5) or the square root of the inverse of the match function (for models 1-4). The square root was used to reduce the difference in fitness between models and thus has the effect of lowering the selection pressure. This was found to be useful in models 1-4 since there was often a very wide range of match values, and if selection is too strong premature convergence of the algorithm

of the ith spike in trace n of the model data, N is the total number of spikes in all traces, the first sum is over the traces and the second sum is over the spike times in the individual traces. $\Theta(i,n)$ equals 1 if both traces n have spike i; otherwise it is 0. The other terms are:

$$A_{i,n} = (max(0.010,\ o_{i,n}))^{pow1}$$

$$B_{i,n} = (1 - \Theta(i,n)) \cdot \frac{P}{(TC_n)^{pow2}}$$

$$C_n = Q \cdot \sum_{i=2}^{N-1} h(ISI_{i-1} - ISI_i)$$

Term A weights spike mismatches more heavily the closer they are to the start of the trace (0.010 is in seconds and time 0 is the beginning of current injection). $pow1$ is a free parameter which can control how strong the weighting is. We used a value of 0.4 for $pow1$. Term B penalizes missing or extra spikes in the simulated trace relative to the target trace. P is a penalty parameter which we set to 0.5 sec. TC_n is the total count of spikes in target trace n. $pow2$ is a free parameter which we set to 0.6. For a $pow2$ value of 0, all extra or missing spikes are penalized the same in all cases; for a higher value of $pow2$ the penalty is reduced when there are a large number of spikes in the target trace. This prevents a single extra spike near the end of a waveform from causing a large penalty when the trace has many spikes already. Term C is a non-monotonicity penalty that penalizes consecutive interspike intervals (ISIs) which are not monotonically decreasing (as they are in the target data sets). The function $h(x)$ is zero if $x < 0$ and is x otherwise. The parameter Q is a penalty value for non-monotonically decreasing ISIs which we set to 200.0 sec^{-1}.

2.4.2 Waveform match function

The waveform match was calculated as:

$$match = \frac{1}{M \cdot N} \sum_{i}^{M} \sum_{i}^{N} (t_i - o_i)^2$$

where o_i and t_i represent respectively the target and test waveforms sampled at 0.1 $msec$, N was the total number of points in a each waveform, and M is the total number of waveforms generated by the model.

known that passive properties measured in whole-cell recordings can be significantly different from those for sharp-electrode recordings, primarily due to the relative leakiness of the soma membrane in the latter caused by impalement damage (Staley et al., 1992). Second, the firing pattern of the cells used to parametrize the original model were quite different from the data used for this model. Finally, the passive structure of the simplified model is clearly different from the original model. However, using the parameters from a complex model as a starting point is a useful strategy to follow in parametrizing a simpler model.

2.3 Target data sets

For models 1 to 4 we used artificial data sets generated from the models themselves *i.e.* we chose values of the parameters that would give desired behaviors, generated outputs as membrane potential waveforms or lists of spike times of the models given particular inputs, and used those outputs as our data sets. For model 5, the data was derived from brain-slice experiments on rat piriform cortex. A layer 2 superficial pyramidal cell's responses to varying levels of current injection were recorded with standard whole-cell intracellular recording techniques. We took data from six different values of current injection (spanning the range from a level at which the cell did not spike at all to a level at which the cell spiked fairly rapidly). The data was processed to extract the spike times; this was the data set that our model had to match. The original inputs consisted of one-second current pulses; to speed up our simulations we only injected currents for 300 msec and used the output spike times for our parameter searches.

2.4 Computing the match between data and models

We used two different match functions to assess how well our models matched our data. For models 1, 2, and 5 (spiking cell models) we processed the membrane potential traces to generate lists of spike times, and then used a match function to compare the spike times of the model with those of the data set. For models 3 and 4 (nonspiking dendritic models) the match function computed the mean squared difference at each time point of the membrane potentials of the model output *vs.* the target data. The match functions are error functions, with 0 representing a perfect match.

2.4.1 Spike timing match function

This matching function computes a weighted average absolute difference in corresponding spike times between the two models, with some extra penalty terms. The spike timing match function is as follows:

stead of having it vary based on the calcium concentration, and by modeling the intracellular calcium concentration as being derived from a leaky integrator whose input is the current through the calcium channel (suitably scaled to change current to calcium concentration).

The ionic conductances present in the cell included fast Na, delayed-rectifier K, inactivating K (A-current), non-inactivating slow K (M-current), high-voltage-activated Ca, non-inactivating (persistent) Na, and two distinct calcium-dependent K (AHP or afterhyperpolarization) channels. One calcium-dependent K channel has a much smaller time constant ($tau([Ca])$) than the other; we refer to this channel as the fast AHP channel. Although there is no direct evidence demonstrating this channel's presence in this cell type, we found that including this channel improved the results of the parameter search significantly (see below). Most of the kinetic parameters were taken either from existing data for piriform cortex superficial pyramidal cells (Banks et al., 1996; Constanti et al., 1985; Galvan et al., 1985; Halliwell and Scholfield, 1984), or, in cases where no such data existed, from other cell types such as hippocampal pyramidal cells (French et al., 1990; McCormick and Huguenard, 1992; Traub et al., 1991). Further details on the model can be found in the appendix. The model thus contained eight voltage- or calcium-dependent ionic conductances and a simple calcium buffer, with 23 parameters chosen for fitting. In addition to the parameter types used for models 1-4, model 5 also includes parameters of the following types:

1. voltage offsets for the activation ($m_\infty(V)$ or $m_\infty([Ca])$) curve of an ionic channel gating variable

2. parameters controlling the calcium buffer model

3. voltage offsets for resting membrane potential

4. somatic and dendritic membrane resistivities

Parameters that are offsets from a starting value (*i.e.* not scale parameters) are chosen from a uniform distribution, as opposed to the log-uniform distribution chosen for the scale parameters.

We added a separate parameter for the resting membrane potential because the values obtained from our experiments included an unspecified junction potential which could be as high as several millivolts (Zhang and Krnjević, 1993). We also considered the specific membrane resistivity of the soma and dendritic regions as separate parameters. Since the reconstructed morphology we used did not correspond to the cell from which the data was taken, allowing membrane resistivities to vary allows us to compensate for this to some extent.

optimal match as with models 1 and 2.

Model 3 was a 100-compartment linear unbranched passive dendrite model. We assumed that RA, RM, and CM were constant for all compartments in the cable but were unknown; thus, we had three parameters to fit. Current was injected at one end of the dendrite. We measured the membrane potential at three points along the dendrite and used this as our output data set to be matched to the target model.

Model 4 was a branched dendrite model; in addition to the soma we had four linear branches, each with 24 compartments. We set the target values of the passive parameters RA, RM, and CM to be significantly different for each of the four branches and the soma in order to generate waveforms at the end of each branch which differed substantially from each other. Input current was injected into the soma. The outputs of the model consisted of the membrane potentials at the soma and those at the distalmost compartments of all four branches. The parameters for this model were RA, RM, and CM for each branch (these parameters were kept uniform within a branch) and for the soma, giving a total of 15 parameters.

It should be noted that for models 3 and 4 we measure the membrane potential at multiple points in the dendrites, which is very difficult to do reliably in real experiments. Similarly, in model 4 the passive parameters were not considered to be fixed over the entire dendrite. Therefore, finding matches for these passive models is considerably more difficult than if the membrane potential had only been measured at a single point (*e.g.* the soma) and if the passive parameters were considered to be uniform over the entire dendrite. The latter case is much more representative of currently existing passive neuron models (*e.g.* (Holmes and Rall, 1992a; Holmes and Rall, 1992b; Rall et al., 1992; White et al., 1992)); however, for this type of model excellent analytical techniques exist to determine parameters (Jack et al., 1983; Koch and Segev, 1989; Koch and Segev, 1997; Rall, 1995), and therefore parameter searches are unnecessary. Our purpose in using the more complex models is to demonstrate the efficacy of parameter searches on passive models for which no good analytical techniques exist. Even though such models may be beyond the current experimental state of the art, advances in multi-dendrite recordings (*e.g.* (Buzsaki and Kandel, 1998; Magee and Johnston, 1995)) will make such models more relevant in the future.

2.2.3 Model 5: layer 2 superficial pyramidal cell in piriform cortex

This model was a simplification of a large model derived directly from an anatomical reconstruction of a layer 2 superficial pyramidal cell from piriform cortex (Koch and Segev, 1997). The original model consisted of 503 compartments and included a number of voltage- and calcium-dependent ionic conductances. The active conductances of this model were restricted to the soma, because there is currently little

channels were derived from a model of the hippocampus by Roger Traub (Traub et al., 1991) while the last was derived from a model of a bullfrog sympathetic ganglion cell (Koch and Segev, 1989). The kinetics of the channels were modified somewhat in order to produce a firing pattern similar to that of regular-spiking pyramidal cells in cerebral cortex (McCormick et al., 1985). In the first model (simple1) the free parameters represented the maximal conductances of the four ionic channels. The second version (simple2) added parameters that scaled the $\tau(V)$ curves (time constant as a function of membrane potential) of the activation gating variable of each of the channels. The first model thus had four adjustable parameters while the second had eight.

Before the searches were started, the initial values of the parameters to be varied were set to random values drawn from a log-uniform distribution centered around the correct values of the parameters. A random value from this distribution is equal to the exponential of a value drawn from a distribution uniformly distributed between the natural logarithms of the endpoints. This distribution allows us to choose random values for parameters with equal probabilities of choosing values of $start/x$ and $start * x$. This is a natural choice for nonnegative scale-dependent parameters such as maximal conductances and scaling values for $\tau(V)$ curves. The range of the distribution was from $\frac{1}{4}$ to 4 times the correct value. We generally chose the ranges for a given parameter by trial and error unless there was experimental data to constrain the range. For models 1 and 2 a range of 4 was sufficient because the initial parameter values were constrained to be no more than a factor of 4 from the correct values. For these models, the goal of the parameter searches was to match the spike timings of the target model's outputs; the shape of the interspike interval waveform was not considered. The inputs were somatic DC current injections (current-clamp) at six different current levels. The reason for using multiple current levels is that in our experience matching a model to data using only a single value of injected current is relatively easy, while matching a model using a wide range of input currents is extremely difficult because of the highly nonlinear behavior of neurons. Thus it is essential to use as much data to constrain the model's behavior as is feasible, and forcing the model to match the data over a wide range of input currents has proven to be an easy way to strongly constrain the parameter values (see the Discussion).

2.2.2 Models 3 and 4: passive dendrite models

Models 3 and 4 (known as *passive1* and *passive2* respectively) were passive dendrite models *i.e* they had no voltage- or calcium-dependent ionic conductances. Our approach was to choose a fixed morphology for the model (presumably known to the modeler) and consider the passive electrical properties of the model (membrane axial resistivity (RA), specific membrane capacitance (CM) and specific membrane

principle be applied to this problem. However, an assessment of the value of one or another automated searching technique can only come from a rigorous side-by-side comparison. In this paper, we specifically evaluate four common searching algorithms which have been applied to neural models: conjugate gradient-descent (Bhalla and Bower, 1993), stochastic-search (Foster et al., 1993), genetic algorithms (Baldi et al., 1998; Eichler-West and Wilcox, 1997; Vanier and Bower, 1996), and simulated annealing (Baldi et al., 1998; Vanier and Bower, 1996). We compare the effectiveness of these methods in obtaining good parameter sets given the same amount of computer resources. Since it is often the case that the effectiveness of different searching algorithms depends on features of the models whose parameters are being optimized, we compare the performance of these methods on five different single-neuron models ranging from very simple to fairly complex, in order to see what general conclusions can be drawn in order to guide modelers in their choice of parameter-searching methods. In the discussion we consider the benefits and limitations of these parameter-searching techniques as well as how these techniques can be used most effectively by neural modelers.

2 Methods

2.1 Simulation environment and data analysis tools

The compartmental neural simulation program GENESIS (Bower and Beeman, 1995), version 2.0, was used for all of the simulations described here. Parameter-search routines and various matching functions were implemented as an extension library in GENESIS. Simulations were run using the implicit Crank-Nicolson method (Bower and Beeman, 1995; Koch and Segev, 1989) with a time step of 20 μsec. Outputs were saved to disk every 0.1 $msec$. Simulations were run on a variety of workstations running Linux or Solaris 2.3 (Sun Microsystems). Data analysis was done using Matlab (The MathWorks, Inc.) and with a variety of scripts written in the Python computer language (http://www.python.org).

2.2 Models

Five models were used to evaluate different parameter-search methods. For any given model, the parameter-search routines were run for a fixed number of simulations and the progress of the search was recorded as the simulation progressed. The purpose of this was to see which search methods gave the best results for a fixed number of simulations. Models 1-4 were used only to test the parameter search methods and are not realistic models of actual neurons, whereas model 5 is based on real data.

perimental data, such as transmembrane voltage trajectories and detailed neuronal spiking patterns generated experimentally in response to complex stimuli. Typically, such models include every voltage- and calcium-dependent ionic conductance known to exist in a particular neuron type, and their morphologies are based upon anatomical data collected from real neurons.

Ideally, all the constituents of the modeled neuronal structures should be derived directly from experimental data. In practice, this is usually not possible for several reasons. It is often the case that the kinetics of many of the candidate ionic conductances have not been adequately characterized experimentally. The spatial distribution and channel densities of the included ionic conductances are often unknown, because these values are relatively difficult to measure experimentally. Models may also incorporate intracellular calcium and second-messenger dynamics (Koch and Segev, 1997), which may not be sufficiently well characterized to completely constrain the model. For these reasons, realistic neuronal modelers are usually faced with constructing simulations containing a large number of loosely-constrained parameters under expectations that the models will replicate complex experimental data with high accuracy.

Given the complexity of the models and the lack of sufficient experimental data, the traditional way in which parameters are assigned to neural models involves varying parameters iteratively by hand based on a series of educated guesses. However, the large numbers of parameters involved (which is steadily increasing as models become more sophisticated) makes manually parameterizing a model a very time-consuming and tedious exercise. In addition, ionic conductances in neural models often interact with one another in highly nonlinear ways (Koch and Segev, 1989; Rinzel, 1985), making it impossible to accurately adjust parameters independently of each other. Manual parameterization also does not provide any quantification of how closely the model replicates physiological data. Instead, typically several figures showing similar waveforms and spiking patterns has sufficed as proof of biological relevance. In practice, this means that modelers are always tempted to stop when the model produces a barely adequate fit to the data, instead of trying to determine the best possible match between the model and the data.

For these reasons, we (Baldi et al., 1998; Bhalla and Bower, 1993; Vanier and Bower, 1996) and others (Eichler-West and Wilcox, 1997; Foster et al., 1993; Tawfik and Durand, 1994) have experimented with the use of automated parameter-search methods for assigning parameters to neural models. This approach requires the modeler to have a target data set that the model must match. A simulation is run, a numerical match value is calculated, and the parameter-search routine automatically chooses the next set of parameters to evaluate based on the previous results and the match value. While, in theory, such a procedure could be iterated over the

Parameter Search Methods for Compartmental Neural Models

Michael C. Vanier*
Department of Computation and Neural Systems
California Institute of Technology
Pasadena, CA 91125
mvanier@bbb.caltech.edu
(626) 395-6818

James M. Bower
Department of Computation and Neural Systems
California Institute of Technology
Pasadena, CA 91125
jbower@bbb.caltech.edu
(626) 395-6817

Abstract

One of the most difficult and time-consuming aspects of building compartmental models of single neurons is assigning values to free parameters in order to make models match experimental data. Automated parameter-search methods potentially represent a more rapid and less labor-intensive alternative to choosing parameters manually. Here we compare the performance of four different parameter-search methods on several single neuron models. The methods compared are conjugate-gradient descent, genetic algorithms, simulated annealing, and stochastic search. Each method has been tested on five different neuronal models ranging from simple models with between 3 and 15 parameters to a realistic pyramidal cell model with 23 parameters. The results demonstrate that genetic algorithms and simulated annealing are generally the most effective methods. Simulated annealing was overwhelmingly the most effective method for simple models with small numbers of parameters, but the genetic algorithm method was equally effective for more complex models with larger numbers of parameters. The discussion considers possible explanations for these results and makes several specific recommendations for the use of parameter searches on neuronal models.

* To whom all correspondence should be addressed.