

Modeling extracellular fields for a three-dimensional network of cells using NEURON

Appukuttan, Shailesh; Brain, Keith L; Manchanda, Rohit

DOI:

[10.1016/j.jneumeth.2017.07.005](https://doi.org/10.1016/j.jneumeth.2017.07.005)
[10.1016/j.jneumeth.2017.07.005](https://doi.org/10.1016/j.jneumeth.2017.07.005)

License:

Creative Commons: Attribution-NonCommercial-NoDerivs (CC BY-NC-ND)

Document Version

Peer reviewed version

Citation for published version (Harvard):

Appukuttan, S, Brain, KL & Manchanda, R 2017, 'Modeling extracellular fields for a three-dimensional network of cells using NEURON', *Journal of Neuroscience Methods*, vol. 290, NSM_7781, pp. 27-38.
<https://doi.org/10.1016/j.jneumeth.2017.07.005>, <https://doi.org/10.1016/j.jneumeth.2017.07.005>

[Link to publication on Research at Birmingham portal](#)

General rights

Unless a licence is specified above, all rights (including copyright and moral rights) in this document are retained by the authors and/or the copyright holders. The express permission of the copyright holder must be obtained for any use of this material other than for purposes permitted by law.

- Users may freely distribute the URL that is used to identify this publication.
- Users may download and/or print one copy of the publication from the University of Birmingham research portal for the purpose of private study or non-commercial research.
- User may use extracts from the document in line with the concept of 'fair dealing' under the Copyright, Designs and Patents Act 1988 (?)
- Users may not further distribute the material nor use it for the purposes of commercial gain.

Where a licence is displayed above, please note the terms and conditions of the licence govern your use of this document.

When citing, please reference the published version.

Take down policy

While the University of Birmingham exercises care and attention in making items available there are rare occasions when an item has been uploaded in error or has been deemed to be commercially or otherwise sensitive.

If you believe that this is the case for this document, please contact UBIRA@lists.bham.ac.uk providing details and we will remove access to the work immediately and investigate.

Modeling Extracellular Fields for a Three-Dimensional Network of Cells using NEURON

Shailesh Appukuttan^{1,*}, Keith L. Brain², Rohit Manchanda¹

¹ Department of Biosciences and Bioengineering,
Indian Institute of Technology Bombay, Mumbai, Maharashtra, India

² School of Clinical and Experimental Medicine, College of Medical and Dental Sciences,
University of Birmingham, Birmingham, UK

Supplementary Document

S1. Toy Neuron Model

We present here further details about the development of the toy neuron model (see section 3.5). NEURON codes for model development and for the linking of extracellular spaces, via the technique presented in this work, have been provided along with a flowchart that summarizes the various steps involved.

S1.1. Model Biophysics

As seen in Fig. 13, each neuron consists of a soma, axon, and two proximal dendrites, each of which divides into two distal dendrites. The soma is spherical and is attached to the axon at one end, and the two proximal dendrites at the other end. The axon has been modeled as being long and thin, with uniform diameter. The diameter of the proximal and distal dendrites is tapered with increasing distance from the soma. The morphological parameters for the neuron model are presented in table S1. The soma and the axon were endowed with Hodgkin-Huxley (HH) channels so as to enable them to produce action potentials. The dendrites were provided only passive leak channels. The reversal potential of the leak channels was set to -65 mV, similar to that for the HH

*Current Address: Neuroinformatics group, Unité de Neurosciences, Information et Complexité, Centre National de la Recherche Scientifique, Gif sur Yvette, France

Table S1: Morphological parameters for the toy neuron model

Section	Parameter	Units	Value
soma	length	μm	20
	diameter	μm	20
	# of segments	-	1
axon	length	μm	1000
	diameter	μm	5
	# of segments	-	1001
proximal dendrites	length	μm	200
	diameter	μm	2 \rightarrow 1
	# of segments	-	201
distal dendrites	length	μm	100
	diameter	μm	1 \rightarrow 0.2
	# of segments	-	101

25 channels, thereby resulting in a resting membrane potential of around -65 mV for the
26 neuron.

27

28 ***S1.2. NEURON Code For Creating The Individual Neurons***

29 The NEURON code presented below constructs a cell template, based on the biophysical
30 parameters specified in the previous section. This template is then utilized to create two
31 instances of the toy neuron.

```
32 // ***** Start of Template *****  
33 begintemplate dummy_neuron  
34     public soma, axon, p_dend, d_dend  
35     create soma, axon, p_dend[1], d_dend[1]  
36  
37     proc init() {
```

```

38     create soma, axon, p_dend[2], d_dend[4]
39     soma {
40         L = 20
41         nseg = 1
42         diam = 20
43         insert hh
44     }
45     axon {
46         L = 1000
47         nseg = 1001
48         diam = 5
49         insert hh
50     }
51     forsec "p_dend" {
52         L = 200
53         nseg = 201
54         diam(0:1) = 2:1
55         insert pas
56         e_pas = -65
57     }
58     forsec "d_dend" {
59         L = 100
60         nseg = 101
61         diam(0:1) = 1:0.2
62         insert pas
63         e_pas = -65
64     }
65
66     //Connecting all the sections together
67     //-- Axon to Soma
68     connect axon(1), soma(0)
69     //-- Primary Dendrites to Soma
70     for i = 0, 1 {
71         connect p_dend[i](0), soma(1)
72     }
73     //-- Secondary Dendrites to Primary Dendrites
74     for i = 0, 1 {

```

```

75         connect d_dend[i*2](0), p_dend[i](1)
76         connect d_dend[(i*2)+1](0), p_dend[i](1)
77     }
78 }
79 endtemplate dummy_neuron
80 // ***** End of Template *****
81
82 //Creating the Neurons
83 objref neuron[2]
84 for i = 0, 1 {
85     neuron[i] = new dummy_neuron()
86 }

```

87

88 ***S1.3. NEURON Code For Linking The Extracellular Fields***

89 The NEURON code below implements the coupling of extracellular fields of the two
90 adjacent neurons, at regions where they are in close spatial proximity.

```

91 objref slist1, slist2, list_temp1, list_temp2
92 objref gmat, cmat, bvec, e, xl, layer, sl, lm
93
94 RATIO_ra_by_re = 0.01 // Specify
95 Re = Ra/RATIO_ra_by_re // Ohm.cm
96 xradial_value = Re*1e-6/(PI*((diam/2)^2)*1e-8) // MOhm/cm
97
98 Re_Ohm = Re*1e-6*L*1e-4/(PI*((diam/2)^2)*1e-8) // MOhm
99 rlink = Re_Ohm/nseg // MOhm
100 glink = 1000/rlink // nS
101 ge_value = (glink*0.1)/area(0.5) // S/cm2
102
103 proc setExtra() {
104     forall {
105         insert extracellular
106         xc[0] = 0
107         xc[1] = 0

```

```

108
109         xg[0] = 1e-9    // Infinite Resistance
110         xg[1] = 1e-9    // Infinite Resistance
111
112         xraxial[0] = xraxial_value           // MOhm/cm
113         xraxial[1] = 1e9    // Infinite Resistance
114     }
115
116     print " _____"
117     print "Extracellular Mechanism Inserted"
118     print " _____"
119 }
120 setExtra()
121
122 proc setExtraLink() {
123     // connected sections in neuron 1
124     slist1 = new SectionList()
125     // connected sections in neuron 2
126     slist2 = new SectionList()
127
128     // ensure that the order is the same
129     neuron[0].axon slist1.append()
130     neuron[1].axon slist2.append()
131     neuron[0].soma slist1.append()
132     neuron[1].soma slist2.append()
133     neuron[0].p_dend[0] slist1.append()
134     neuron[1].p_dend[0] slist2.append()
135     neuron[0].d_dend[0] slist1.append()
136     neuron[1].d_dend[0] slist2.append()
137     neuron[0].d_dend[1] slist1.append()
138     neuron[1].d_dend[1] slist2.append()
139
140     nsegs = 0 // will contain total connected segs
141     forsec slist1 {
142         nsegs += nseg
143     }
144     print " _____"

```

```

145 print "Total Connected Segments = ", 2*nsegs
146 print " _____"
147
148 gmat = new Matrix(2*nsegs, 2*nsegs, 2)
149 cmat = new Matrix(2*nsegs, 2*nsegs, 2)
150 bvec = new Vector(2*nsegs)
151 xl = new Vector()
152 layer = new Vector(2*nsegs)
153 layer.fill(1)
154
155 forsec slist1 {
156     for (x, 0) {
157         xl.append(x) // for neuron 1
158         xl.append(x) // for neuron 2
159     }
160 }
161
162 e = new Vector(2*nsegs)
163 sl = new SectionList()
164
165 list_temp1 = new List()
166 forsec slist1 {
167     for (x, 0) {
168         list_temp1.append(new SectionRef())
169     }
170     slist1.remove()
171 }
172 list_temp2 = new List()
173 forsec slist2 {
174     for (x, 0) {
175         list_temp2.append(new SectionRef())
176     }
177     slist2.remove()
178 }
179
180 for i = 0, nsegs-1 {
181     list_temp1.object(i).sec sl.append()

```

```

182         list_temp2.object(i).sec sl.append()
183     }
184
185     ge = ge_value // S/cm2
186
187     for (i=0; i<(2*nsegs); i=i+2) {
188         gmat.x[i][i] += ge
189         gmat.x[i+1][i+1] += ge
190         gmat.x[i][i+1] += -ge
191         gmat.x[i+1][i] += -ge
192     }
193
194     lm = new LinearMechanism(cmat, gmat, e, bvec, sl, xl,
195 layer)
196
197     print "_____ "
198     print "Extracellular Connected Via Link"
199     print "_____ "
200 }
201 setExtraLink()
202

```

203 As described in section 3.5, the neurons have been oriented such that one primary
204 dendrite of each of the two neurons, along with the distal dendrites emerging from
205 them, are considered to be at a sufficiently large distance from the other neuron to not
206 have their extracellular spaces affected directly by it. This constraint was imposed to
207 demonstrate an element of heterogeneity in the coupling of the extracellular regions of
208 the two neurons. The exact nature of coupling between the two neurons can be made as
209 complex as necessary, with the technique being amenable to any such requirements.
210 The toy neuron model simply being an illustrative example, we have chosen to limit its
211 complexity here by assigning equal coupling strengths between the various regions at
212 which the two neurons are in close spatial proximity. This assumes that the two

213 neurons maintain a constant spatial separation throughout their morphology. If the
214 need arises for differential coupling, potentially based on the distance of separation
215 and/or other factors, the implementation requires only a minor modification to reflect
216 the same. In the above implementation, this would translate into the dynamic
217 evaluation of the parameter '*ge_value*' for each segment, with the value being a function
218 of the distance of separation.

219 In the case of a network/syncytial model, variations in the boundary conditions are
220 easily feasible within the presented framework. The extracellular boundary conditions,
221 in NEURON, can be altered as required by modifying the values of the resistive,
222 capacitive and voltage components (see Table 1 and Fig. 2) that define the extracellular
223 features of each cell. Variations can be introduced, such as between peripheral and non-
224 peripheral cells, or across the ends of a tissue/network, depending on the modeling
225 requirements.

226 ***S1.4. Flowchart Describing Steps Followed For Toy Model***

227 Fig. S1 shows a flowchart that summarizes the series of steps involved in developing the
228 toy model. The steps involved in the coupling of the extracellular spaces of the two
229 neurons, which form the crux of the present work, have been encapsulated within the
230 dashed box. The only main prerequisite is to have the models of the individual neurons
231 that need to be coupled extracellularly. The various stages, and also the terminology
232 used for the various parameters (in red), are closely linked to the NEURON code
233 presented in the previous section. Certain stages in the workflow have been
234 represented via colored boxes, and these correspond to the stages where user input is
235 required. The other stages are capable of continuing in their default configuration,

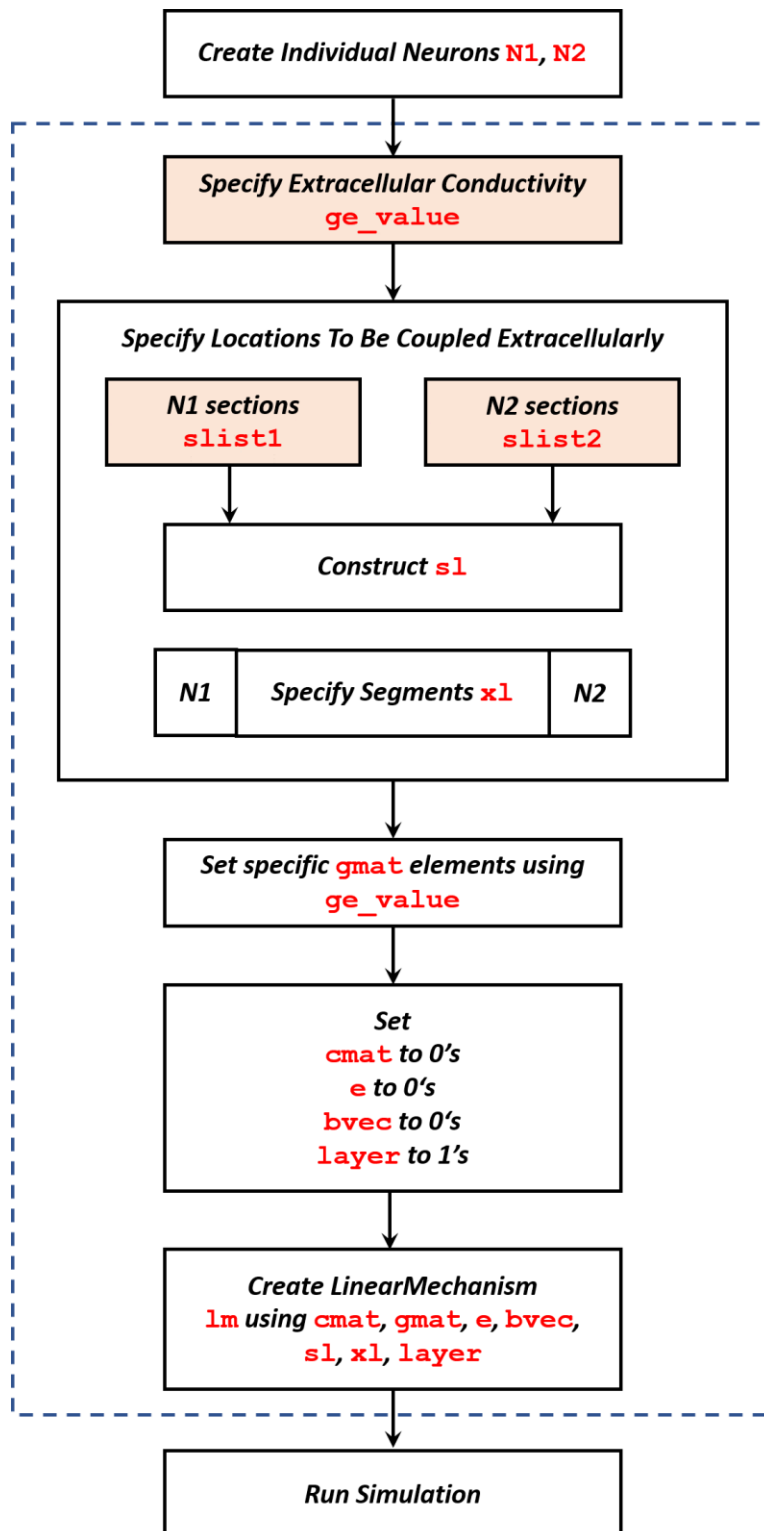


Figure S1: Flowchart summarizing the steps involved in developing the toy model. The dashed box encapsulates the steps involved in the coupling of the extracellular fields, and corresponds to the NEURON code presented in section S1.3. The colored boxes indicate the minimal stages where user input is required. The terms in red correspond to the variable names employed in the NEURON code.

237 coupling. This offers a degree of automation if the modeler simply wishes to reuse the
238 existing extracellular coupling mechanism between different pairs of neurons. The
239 process can be further streamlined by creating templates for the coupling mechanism
240 and instantiating these wherever required. This would, in essence, be similar to creating
241 templates for cells as employed in section S1.2, and may be advisable when handling
242 large neuronal networks. But the wider capabilities and flexibility of the technique
243 presented here is more evident when the modeler opts to explicitly tweak the individual
244 parameters as dictated by the biophysical requirements of their model. The flowchart
245 only attempts to present a very basic and easily reusable implementation of the
246 technique.