Brian²

Github repository: https://github.com/brian-team/brian2
Documentation: https://brian2.readthedocs.org

Main syntax changes compared to to Brian 1:

NeuronGroup	
No “variable guessing” for resets, thresholds and refractoriness	`G = NeuronGroup(100, 'dv/dt = -v/tau: 1 (unless-refractory)',` `threshold='v>1', reset='v=0', refractory=5*ms)`

More flexible refractoriness mechanism	`refractory=5*ms  # fixed time`
	`#one time per neuron, stored in state variable “ref”:`
	`refractory='ref'`
	`refractory='1*ms+rand()*ms'  # changes after every spike`
	`# stays refractory as long as condition is true:`
	`refractory='v>v_th'`

State variable assignment with strings possible	`G.v = 'i / 100.0 + 0.05*randn()'`
Equations	
No more aliases, but support for string_replacement	`Equations('dv/dt = g_L*(E_L - v) / tau : volt',` `g_L='g_Le', E_L='E_L_e', tau='tau_e')`
Synapses	

Completely replaces connect class, even for simple connections	`weight = 0.1`
	`S = Synapses(G, G, pre='v+=weight', connect=True,`
	`delay=0.5*ms)`

New method for synapse creation instead of assigning to object	`S = Synapses(G, G, 'w:1', pre='v+=w')`
	`S.connect(True)  # full connectivity`
	`S.connect('i ==j ')  # one-to-one connections`
	`S.connect(True, p=0.1)  # 10% connection probability`
	`S.connect(True, n=2)  # 2 synapses per connection`

Indexing with strings	`S.w['i==j'] = 0.1` `S.w['i!=j'] = 0.5`
	Units
Units are not restricted to scalar values	`[1, 2, 3] * mV` `sqrt(mean((arange(10) * Hz)**2))`

Contributions welcome!

File an issue: https://github.com/brian-team/brian2/issues
Fork and submit a pull request: https://help.github.com/articles/using-pull-requests
Development guidelines: https://brian2.readthedocs.org/en/latest/developer/guidelines