Pažintis su programiniu paketu NEURON

NEURON - tai programinis paketas, skirtas nervų sistemos modeliavimui. Jis leidžia sukurti sudėtingus neuronų ir neuronų tinklų modelius.

NEURON išsamų aprašymą galima rasti adresu:
http://www.neuron.yale.edu/neuron/ - NEURON puslapis
http://www.neuron.yale.edu/neuron/docs/help/contents.html - NEURON funkcijų aprašymas

Šio laboratorinio darbo metu išmoksime sudaryti neuroną ir patyrinėsime jo pasyvias savybes. Pasyvus neuronas negeneruoja veikimo potencialo (*action potential, spike*). Laboratorinis darbas paruoštas naudojantis David Sterrat pagalba (Edinburgo universitetas, Škotija)

II. Kaip pradėti ir baigti dirbti NEURON paketu

Norėdami pradėti dirbti NEURON paketu, atverkite NEURON5.5 langą ir spustelkite pele

nrngui

Pamatysite du langus:
1) *nrniv* – langas, kuriame galite rašyti NEURON komandas. Tai vadinamasis komandinis HOC langas.
2) NEURON **Main Menu** – leidžia naudotis grafiniu interfeisu

NEURON paketas uþdaromas **Main Menu→ File→ Quit** arba *Ctrl-D* arba parašant HOC lange `quit()`.

NEURON programos raðomos HOC programavimo kalba. Programos saugojamos failuose su išplėtimu *hoc*.

III. Sekcijų kūrimas ir jų parametrų keitimas

NEURON pakete neurono geometrija aprašoma cilindrinėmis sekcijomis, t.y. neuronas suskaidomas į dalis - sekcijas, kur kiekviena sekcija yra cilindro formos. Kiekviena sekcija gali turėti skirtingas morfologines ir elektrines savybes (ilgį, plotį, talpą, varžą, jonų kanalus). Sekcijos dar gali būti dalinamos į segmentus, kurie leidžia padidinti skaičiavimų tikslumą.

Sukursime neuroną, kuris sudarytas tik iš kūno *(soma)* ir neturi dendritų bei aksono. Tai labai supaprastintas modelis. Iš neurofiziologinių eksperimentų yra žinoma, kad neurono kūno skersmuo yra apie 20 μm.

Komandinėje HOC lango eilutėje parašykite:

```
create soma
```

Ši komanda sukuria sekciją *soma*.
Į sukurtą sekciją galima kreiptis:

```
access soma
```

Sukurtos sekcijos parametrai parodomi:

```
soma psection()
```

NEURON suteikia sekcijos parametrų reikšmes pagal nutylėjimą:

Kintamasis	Paaiškinimas	Vertė	Matavimo vienetai
Nseg	Segmentų skaičius sekcijoje	1	
L	Sekcijos ilgis	100	μm
Ra	Specifinė ašies varža (Specific axial resistance)	35.4	Ω cm
Diam	Skersmuo	500	μm
Cm	Specifinė membranos talpa (Specific membrane capacitance)	1	$\mu F/cm^2$

Norėdami pakeisti šių parametrų reikšmes:
1. Kreipiamės į sekciją:
```
access soma
nseg=2
L=20
diam=20
Ra=120
```
2. Arba naudojame "." žymėjimą:
```
soma.nseg=2
soma.L=20
soma.diam=20
soma.Ra=120
```
3. Arba sugrupuojame kintamuosius ir prieš juos parašome sekcijos vardą:
```
soma {
      nseg=2
      L=20
      diam=20
      Ra=120
}
```

Šių parametrų dabar nekeičiame.
Sekcijos parametrus galime išvesti į ekraną:
1. Kreipiamės į sekciją:
```
access soma
print diam
```
2. Arba naudojame "." žymėjimą:
```
print soma.diam
```

Aukščiau minėtos sekcijų savybės yra priskiriamos pagal nutylėjimą, tačiau kitus mechanizmus reikia nurodyti tiesiogiai programoje. Vienas iš tokių mechanizmų yra pasyvūs jonų kanalai. Šie kanalai apsprendžia neurono membranos pasyvias savybes (neuronas negeneruoja veikimo potencialo).

Pasyvūs jonų kanalai į anksčiau pasirinktą sekciją (naudojant `access` komandą) įterpiami komanda:

```
insert pas
```

Jei norime iš karto kreiptis sekciją ir įterpti pasyvius jonų kanalus:

```
soma insert pas
```

Šia komanda mes įvedėme du papildomus sekcijos parametrus:

Kintamasis	Paaiškinimas	Vertė	Matavimo vienetai
g_pas	Specifinis nuotėkio laidumas (Specific leakage conductance)	0.001	S/cm^2
E_pas	Nuotėkio reversinis potencialas (Leakage reversal potential)	-70	mV

Laidumas g yra atvirkštinis dydis varžai g=1/R ir matuojamas Siemens [S] vienetais.
Specifinis laidumas yra laidumas ploto vienetui, matuojamas S/ cm^2.
Specifinė varža yra varža ploto vienetui, matuojama Ωcm^2.
Nuotėkio reversinis potencialas yra toks membranos potencialas, prie kurio nuotėkio srovių suma lygi nuliui.

Specifinė membranos varža Rm gali būti apskaičiuota, jei žinome specifinį nuotėkio laidumą g_pas:

$$Rm=1/ g\_pas=1/0.001\ S/ cm^2 =1000\ \Omega\ cm^2=1\ k\ \Omega\ cm^2$$

Tipiška specifinė membranos varža yra 10 k Ω cm^2, tad randame, kad g_pas=0.0001 S/ cm^2

Pakeičiame šio parametro reikšmę:

```
soma g_pas=0.0001
```

Membranos potencialas pradiniu laiko momentu lygus –70mV:

v_init=-70

IV. Srovės įleidimas į sekciją

Sukūrėme vieną sekciją *soma* su pasyviomis savybėmis. Norėdami panagrinėti šios sekcijos ypatybes, įleisime srovę ir pažiūrėsime, kaip kinta membranos potencialas. Membranos potencialas yra vienas pagrindinių dydžių, charakterizuojančių neurono būseną. Ramybės būsenoje neurono membranos potencialas yra apie –70mV, tačiau daug dažniau jis būna žemesnis (hyperpolarized neuron) arba aukštesnis (depolarized neuron) už ramybės potencialą (resting potential). Tai priklauso nuo įtekančių ir ištekančių srovių.

Sukuriame srovės šaltinį - objektą *stim*, kuris yra *IClamp* tipo ir patalpintas sekcijos *soma* viduryje. Parametras 0.5 gali kisti nuo 0 iki 1.

```
objref stim
soma stim= new IClamp(0.5)
```

Nurodome *stim* reikðmes – amplitudę (nAmperai), vėlinimà (ms), trukmæ (ms).

```
stim.amp=1
stim.del=100
stim.dur=110
```

Šiomis komandomis į sekciją *soma* mes įvesime srovės impulsą, kurio amplitudė 1 nA, jis prasideda po 100 ms nuo proceso pradžios ir trunka 110ms.

V. Programos vykdymas komandiniame HOC lange

Pažiūrėkime, koks yra membranos potencialas eksperimento pradžioje:

```
soma print v
```

Mes norime rasti membranos potencialą po 300ms. Nurodome, kad procesas trunka 300ms:

```
tstop=300
```

Parašyta programa bus įvykdyta įvedus komandą:

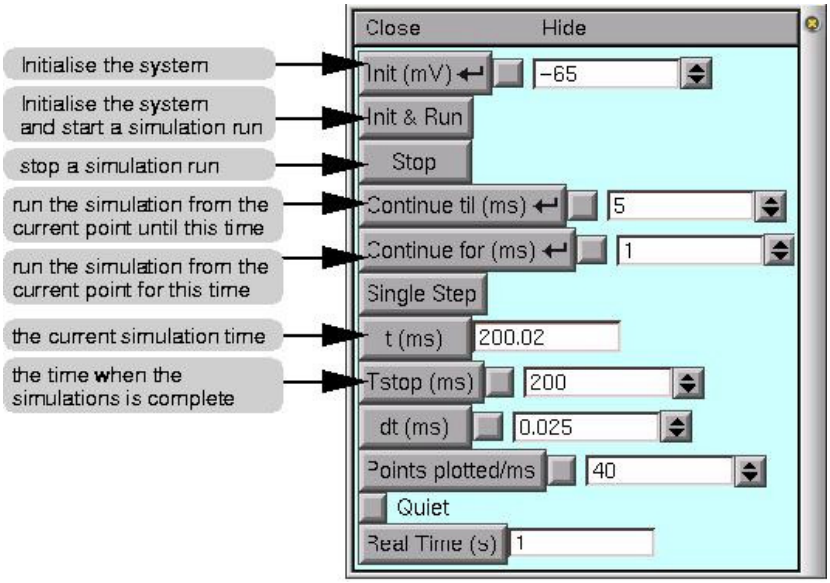
```
run()
```

Pažiūrėkime, kaip pasikeitė membranos potencialas:

```
soma print v
```

VI. Programos vykdymas naudojantis grafiniu interfeisu

Programą galime vykdyti naudodamiesi grafiniu interfeisu **Main Menu**.
Pasirinkime **Main Menu →Tools →RunControl**. Šios komandos atvers langą **RunControl**, kuris leid¸ia nustatyti svarbiausius eksperimento parametrus:



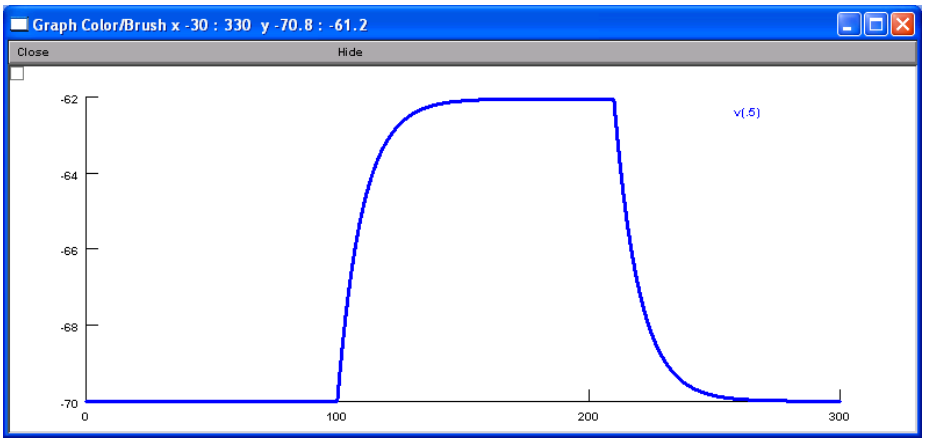
Pav 1. RunControl grafinis interfeisas

Init (mV) nustato pradinę membranos potencialo reikšmę.
Init&Run inicializuoja ir paleidžia vykdyti programą.
Tstop (ms) yra proceso trukmė.
Kitos komandos paaiškintos paveiksle 1.

VII. Rezultatų grafinis vaizdavimas
Rezultatus galime pavaizduoti grafiškai naudodamiesi grafiniu interfeisu.

Membranos potencialo grafinis vaizdavimas

Main Menu →Graph →Voltage axis atveria langą, kuriame bus vaizduojamas membranos potencialas. Jei turėsime ne vieną sekciją, o kelias (kai prijungsime dendritus ir aksoną), reikės nurodyti, kokios sekcijos potencialą brėžti. Galima atverti daug tokių langų.
Paleiskite programą vykdyti **RunControl →Init&Run** ir pažiūrėkite, kaip kinta membranos potencialas.
Norėdami padidinti gautą grafiką, pele užeikite ant **Graph** lango ir dešiniuoju pelės klavišu atverkite papildomą meniu, kuriame pasirinkite **View →View=plot**
Kai padidinsite grafiką, turėtumėte pamatyti membranos potencialą, panašų į paveiksle 2 pavaizduotąjį.



Pav. 2. Membranos potencialo (matuojamo mV) priklausomybė nuo laiko (matuojamo ms).

Srovių grafinis vaizdavimas

Main Menu →Graph →Current axis atveria langą *Graph*, kuriame bus vaizduojamos srovės, tekančios per membraną. Turėsime nurodyti, kokias sroves norime brėžti.

1. Pele užeikite ant šio lango ir dešiniuoju pelės klavišu atverkite papildomą meniu, kuriame pasirinkite **Plot what?**
2. Tai atvers dar vieną langą *NEURON*. Du kartus spustelkite ties *soma* pirmame stulpelyje. Antrajame stulpelyje pamatysite kintamųjų sąrašą. Pasirinkite *i_cap(0.5)* (tai membranos talpos srovė sekcijos *soma* viduryje). Tekstinėje eilutėje viršuje pamatysite *soma. i_cap(0.5)*. Paspauskite **Accept** mygtuką. Graph lange pamatysite užrašą *i_cap(0.5)*.

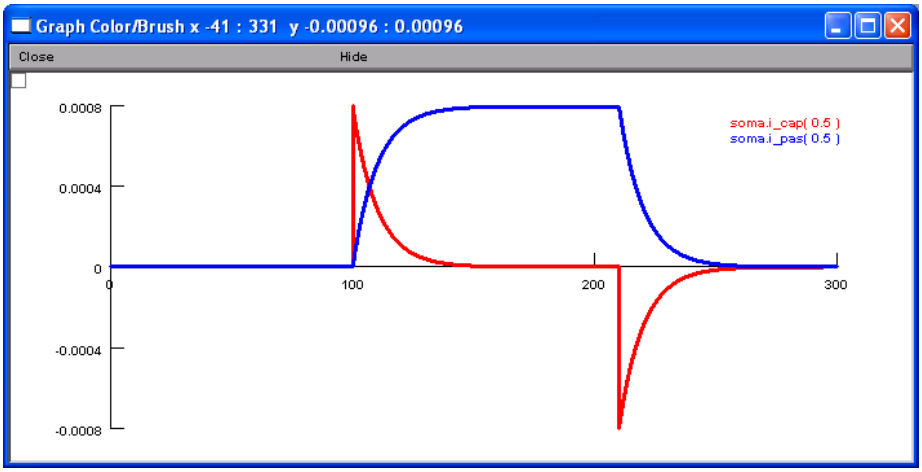
Paleiskite programą vykdyti **RunControl →Init&Run** ir pažiūrėkite, kaip kinta talpos srovė.
Norėdami padidinti gautą grafiką, pele užeikite ant **Graph** lango ir dešiniuoju pelės klavišu atverkite papildomą meniu, kuriame pasirinkite **View →View=plot**

Pavaizduokime ne tik talpos, bet ir nuotėkio srovę viename lange. Pakartosime aukščiau aprašytą procedūrą:

3. Pele užeikite ant šio lango ir dešiniuoju pelės klavišu atverkite papildomą meniu, kuriame pasirinkite **Plot what?**
4. Tai atvers dar vieną langą *NEURON*. Du kartus spustelkite ties *soma* pirmame stulpelyje. Antrajame stulpelyje pamatysite kintamųjų sąrašą. Pasirinkite *i_pas(0.5)* (tai membranos nuotėkio srovė sekcijos *soma* viduryje). Tekstinėje eilutėje viršuje pamatysite *soma. i_pas(0.5)*. Paspauskite **Accept** mygtuką. *Graph* lange pamatysite užrašą *i_pas(0.5)*.

Vėl paleiskite programą vykdyti **RunControl →Init&Run** ir pažiūrėkite, kaip kinta srovės.
Norėdami padidinti gautą grafiką, pele užeikite ant **Graph** lango ir dešiniuoju pelės klavišu atverkite papildomą meniu, kuriame pasirinkite **View →View=plot**.
Norėdami pakeisti vienos iš kreivių storį ar spalvą, pele užeikite ant **Graph** lango ir dešiniuoju pelės klavišu atverkite papildomą meniu, kuriame pasirinkite **Color/brush**. Pasirinkite norimą spalvą ir linijos pobūdį ir

pažymėkite kreivės užrašą (*legend*). Dar kartą paleidę programą, turėtumėte pamatyti sroves, panašias į paveiksle 3 pavaizduotąsias.



Pav. 3. Srovių (matuojamų nA) priklausomybė nuo laiko (matuojamo ms).

VII. Programos rašymas į failą ir jos vykdymas

Iki šiol programą rašėme komandiniame HOC lange. Daug patogiau programas rašyti į failus. Failas turi būti su *hoc* išplėtimu, pvz, *manoNeuronas.hoc*. Programos pradžioje turi būti šios komandos, kurios leidžia atverti **Main Menu** langą:

```
load_file("nrngui.hoc")
load_proc("nrnmainmenu")
```

Programos iš failo vykdymas gali būti atliekamas šiais būdais:

1) Atidarykite NEURON du langus (žr **I** dalį), **Main Menu** menu pasirinkite **Main Menu →File→load hoc**, išrinkite failą ir paspauskite **load**. Programa bus įvykdyta.
2) Spustelkite pele ties failo *manoNeuronas.hoc* vardu, ekrane atsiras du NEURON langai ir programa bus įvykdyta.

Komandas galite rašyti ir į komandinį HOC langą, bet jos nebus išsaugotos. Jei pakeitėte programą, ją perkrauti galite neuždarant NEURON langų:

```
load_file("manoNeuronas.hoc")
```

Mūsų programa dabar atrodo taip:

```
//Loading standard libraries
load_file("nrngui.hoc")
load_proc("nrnmainmenu")
//Soma
create soma
access soma
soma insert pas
soma g_pas=0.0001
soma psection()
//Current source
objref stim
soma stim=new IClamp(0.5)
stim.amp=1 //nA
stim.del=100 //ms
```

```
stim.dur=110 //ms
soma print v
//Time of simulation
tstop=300
```

Pakeiskite sekcijos soma parametrus -

```
    soma {
          nseg=2
          L=20
          diam=20
          Ra=120
    }
```

ir pažiūrėkite, ar pasikeitė membranos potencialas. Kaip manote, kodėl?
Sumažinkite srovės šaltinio amplitudę:

```
stim.amp=0.01
```

VIII. Kelių sekcijų sujungimas

Sukurkime sekciją *axon* ir prijunkime ją prie sekcijos *soma*. Mūsų neuronas taps panašesnis į biologinį neuroną.

```
create axon
access axon
axon insert pas
```

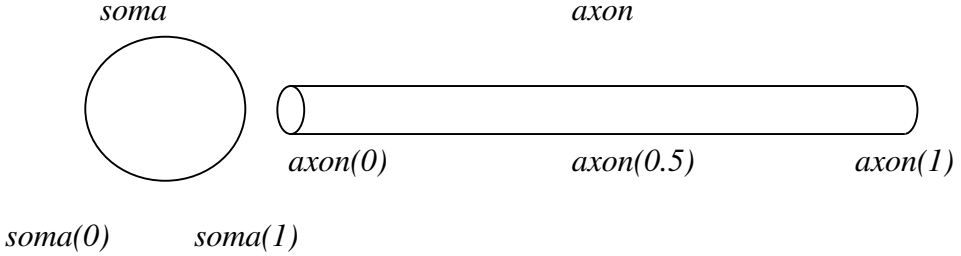
Nurodome šios sekcijos parametrus:

```
axon {
     diam=1
     L=1000
     nseg=50
     g_pas=0.0001
}
```

axon sekcija yra 1μm pločio ir 1000μm ilgio, padalinta į 50 segmentų.

Sujunkime sekcijas *soma* ir *axon*:

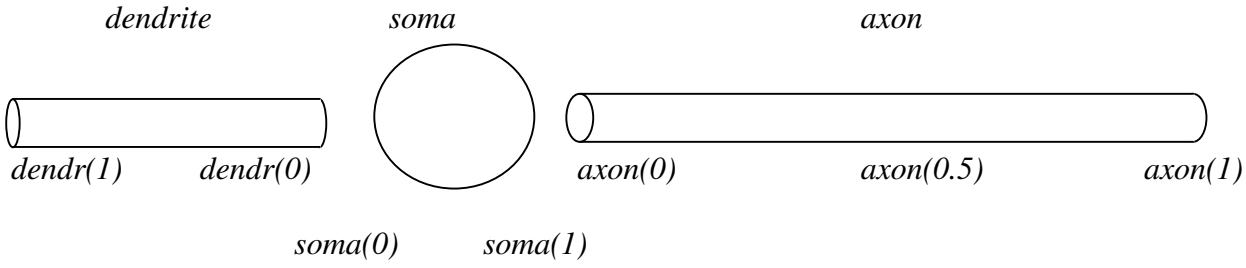
```
connect axon(0), soma(1)
```



sekcijų koorditatės

Skaičiai tarp skliaustelių gali kisti nuo 0 iki 1 ir nurodo sekcijos pjūvio koordinates - atstumą išilgai sekcijos, kuriuo nuo sekcijos pradžios nutolęs išrinktas/ tiriamas sekcijos pjūvis. Sekcijos pradžia laikoma ta sekcijos dalis, kuri yra artimesnė neurono kūnui (*soma*).

Somos, dendrito ir aksono sujungimas atrodytų:



Pav. 4. Sekcijų koordinatės ir sekcijų jungimas.

Sukurkime 2 dendritus ir sujunkime juos su sekcija *soma*:

```
create dend1
dend1 {
     diam=1
     L=500
     nseg=25
     Ra=120
     insert pas
     g_pas=0.0001
}

create dend2
dend2 {
     diam=1
     L=250
     nseg=20
     Ra=120
     insert pas
     g_pas=0.0001
}

//Connecting dendrites and soma
connect dend1(0), soma(0)
connect dend2(0), soma(0)


//looking at topology of the neuron
topology()
```

Sukurto neurono struktūrą galite pažiūrėti:
1) **Main Menu →Graph →Shape plot**. Atsivers langas **Shape**. Galite neuroną apžiūrėti iš visų pusių, t.y. pasukti jį trimatėje erdvėje. Pele užeikite ant **Shape** lango ir dešiniuoju pelės klavišu atverkite papildomą meniu, kuriame pasirinkite **3D Rotate**. Dabar neuroną galite "vartyti" pele.

2) Komandinėje HOC lango eilutėje parašykite:

```
topology()
```

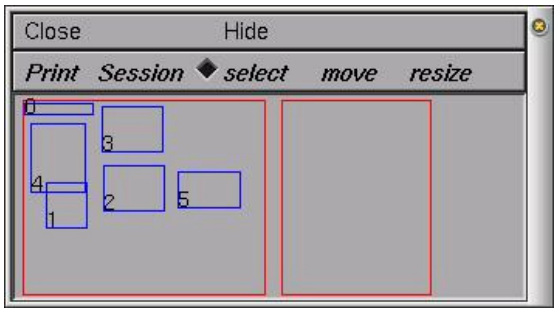
Galite sukurti trečią/ketvirtą dendritą ir prijungti prie pirmojo/antrojo dendrito vidurio, pvz,

```
connect dend3(0), dend1(0.5)
```

X. Rezultatų saugojimas

NEURON langus ir gautus rezultatus (grafikus) galite išsaugoti.

Main Menu →Print&File Widow Manager atveria langą, susidedantį iš dviejų dalių. Kairiojoje dalyje matyti visi NEURON langai. Į dešiniąją dalį galite perkelti pasirinktus langus (**Print&File Window Manager** lango meniu galite naudoti ir komandas *move, resize*, ne tik *select*).



Main Menu →Print&File Widow Manager→Session →Save all
išsaugoja visus langus (kairioji dalis)
Main Menu →Print&File Widow Manager→Session →Save selected
išsaugoja pasirinktus langus (dešinioji dalis)
Failo vardas yra su išplėtimu *ses*. Paprašyti įrašyti failo vardą, nurodykite ir plėtinį, pvz, *manoNeuronas.ses*

Rezultatų failas gali būti pakraunamas
Main Menu →Print&File Widow Manager→Session →Retrieve
arba
programoje ar komandiniame HOC lange parašant

```
xopen("manoNeuronas.ses")
```

XI. Sinapsės

Sukuriame žadinančią sinapsę *synapseExcitatory* ant dendrito *dend1*:

```
objref synapseExcitatory
dend1 synapseExcitatory=new Exp2Syn(0.5)
synapseExcitatory.tau1=1 //synaptic time constant (ms)
synapseExcitatory.tau2=3 //synaptic time constant (ms)
synapseExcitatory.e=0 //synaptic reversal potential (mV)
```

Sinapsė bus sukurta kreipiantis į klasės *ExpSyn* konstruktorių su argumentu 0.5, kuris nurodo sinapsės koordinatę ant dendrito (0.5 – vidurys). Sinapsės laidumo funkcija bus modeliuojama kaip dviejų eksponenčių skirtumas:
`synapseExcitatory.tau1=1` – sinapsės laidumo funkcijos (eksponentės) laiko konstanta, 1ms
`synapseExcitatory.tau2=3` – sinapsės laidumo funkcijos (eksponentės) laiko konstanta, 3ms
`synapseExcitatory.e=0` – sinapsės reversinis potencialas, 0mV

Šie parametrai atitinka AMPA sinapsės parametrus.

b) Sukuriame sinapsės žadinimo šaltinį *spikeSourceExcitatory*, kuris nurodo, kokiais laiko momentais sinapsė yra sužadinima. Sinapsė sužadinama tuomet, kai presinaptinis neuronas generuoja veikimo potencialą (VP), kuris sklinda aksonu ir lemia neurotransmiterio išmetimą į sinaptinį plyšį. Todėl sinapsės žadinimo šaltinis kartu nurodo, kokiais laiko momentais presinaptinis neuronas yra aktyvus.

```
objref spikeSourceExcitatory
dendrite spikeSourceExcitatory = new NetStim(0.5)
spikeSourceExcitatory.interval=200 //ms (mean) time between spikes
spikeSourceExcitatory.number=5 //(average) number of spikes
spikeSourceExcitatory.start=20 //ms (mean) start time of the first spike
spikeSourceExcitatory.noise=0 //  range 0 to 1 - fractional randomness
                              // 0 - deterministic, 1- intervals decaying
                              // Poisson distribution
```

Sinapsės žadinimo šaltinis sukuriamas kreipiantis į klasės *NetStim* konstruktorių. Argumentas šiuo atveju nėra svarbus.
`spikeSourceExcitatory.interval=200 -` (vidutinis) laikas tarp dviejų VP, 200ms
`spikeSourceExcitatory.number=5` – (vidutinis) VP skaičius, 5
`spikeSourceExcitatory.start=20` –(vidutinis) pirmojo VP laiko momentas, 20ms
`spikeSourceExcitatory.noise=0` - kinta nuo 0 iki 1 , 0- deterministinis procesas, 1-VP pasiskirstę Puasono dėsniu

c) Sujungiame sinapsės žadinimo šaltinį *spikeSourceExcitatory* su sinapse *synapseExcitatory*:

```
objref connectionExcitatory
thresh=10 //not important when connection is from NetStim
delay=0.0 //delay of the connection
weight=0.1 //connection strength in muS, maximal conductance
connectionExcitatory=new
NetCon(spikeSourceExcitatory,synapseExcitatory,thresh,delay,weight)
```

Jungtis pavadinta *connectionExcitatory*, ji sukuriama kreipiantis į klasės *NetCon* konstruktorių su argumentais
spikeSourceExcitatory,synapseExcitatory,thresh,delay,weight.

thresh=10 – slenkstis, nėra svarbus naudojant *NetStim*
delay=0.0 – vėlinimas 0ms
weight=0.1 – jungties svoris – maksimalus sinapsės laidumas, 0.1μS.

Sinapsės savybės gali būti keičiamos, pvz:
`spikeSourceExcitatory.interval=100`
`connectionExcitatory.weight=1`