Effect of Atomexitine on Sodium-Channel Inactivation

Michael Fauler

Institute of General Physiology, Ulm University, michael.fauler@uni-ulm.de

01/2019

Table of Contents

Load and Plot Data

Data is arranged in an Excel-Sheet observation-based. Each observation is represented by a row in the table with the following variables (columns):

- ID of observation (ID)
- Sweep/Run ID within experiment (Run)
- Atomexitine concentration (Ato in $\mu g/\mathrm{ml}$)
- Clamp potential (Vp in V)
- normalized current (In)
- Experiment identifier (Exp)

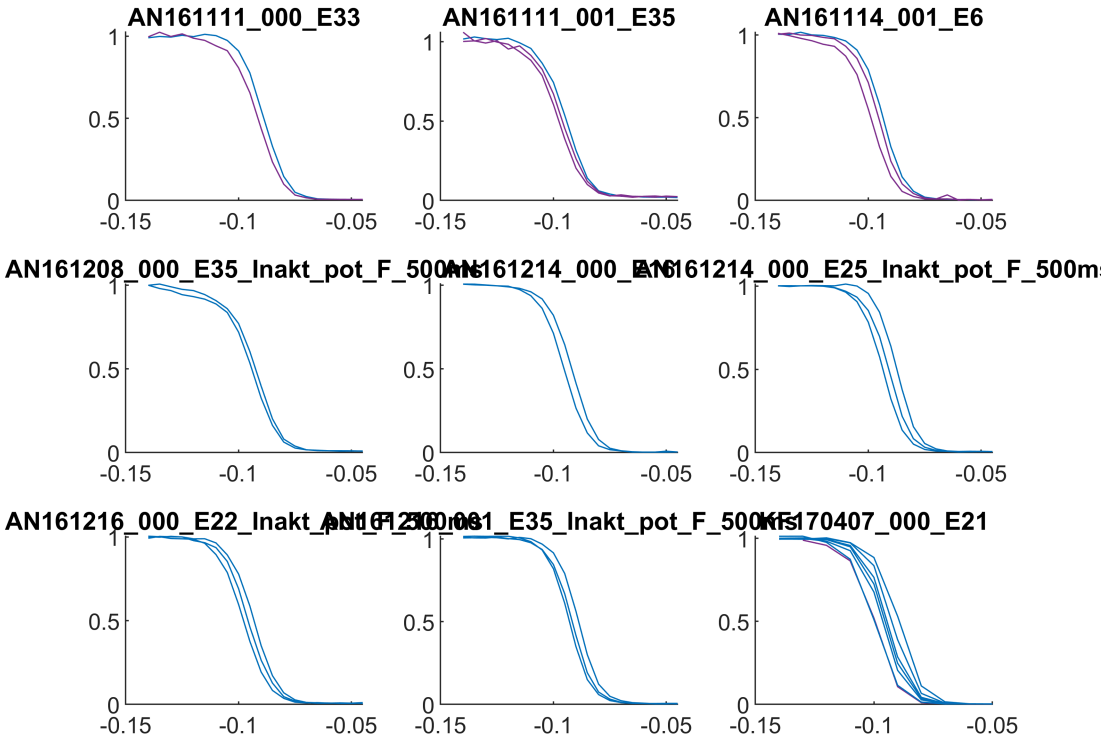
The Excel-Table has been imported into Matlab manually and saved in a *.mat-file.

The plot displays the experimental results. Blue control conditions, magenta if Atomexitine (3 or 10 $\mu g/\mathrm{ml}$) was present.

```
load 'inactivation_data_Ato_shift.mat';
exps = unique(Data.Exp);

figure;
cols = lines(11);
for k=1:numel(exps)
    idx_exp = Data.Exp==exps(k); %observations belonging to experiment exps(k)
    D = Data(idx_exp,:);
    runs = unique(D.Run); %sweeps
```

```
    ato = unique(D.Ato); %Ato concentrations
    subplot(3,3,k);
    hold on;
    for run=1:numel(ato)
        for r=1:numel(runs)
            idx_r = D.Run==r;
            Dr = D(idx_r,:); %Data of that run within experiment exp(k)
            plot(Dr.Vp,Dr.In,'-','Color',cols(Dr.Ato(1)+1,:));
        end
    end
    title(string(D.Exp(1)),'Interpreter','none');
end
```




```
clearvars idx_exp D runs ato c r k idx_r Dr col;
```

There are 9 different experiments and a total of 28 runs.

```
disp(exps);
```

```
    AN161111_000_E33
    AN161111_001_E35
    AN161114_001_E6
    AN161208_000_E35_Inakt_pot_F_500ms
    AN161214_000_E16
    AN161214_000_E25_Inakt_pot_F_500ms
    AN161216_000_E22_Inakt_pot_F_500ms
    AN161216_001_E35_Inakt_pot_F_500ms
    KF170407_000_E21
```

Fit Independent Inactivation Curves

The data has an hierarchical structure (top-down order):

1. experiment
2. various number of runs/sweeps in each experiment
3. single observations (data points) at different clamp potentials (V_p).

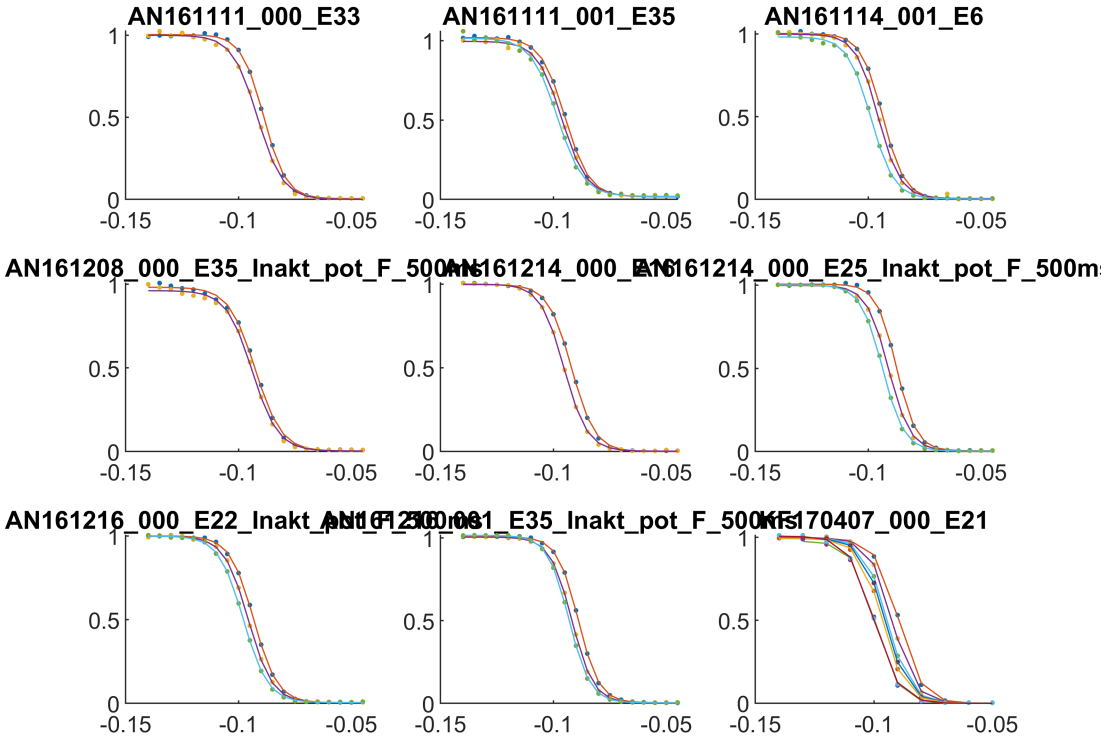
Each sweep is fitted independently. In the next section dependence of parameter values from experiment and/or sweep number is evaluated.

```
b0 = [ -.09, 1, 0, .01 ]; %initial values [ vh, a, i0, k ]
opts = statset( 'Display', 'final', ...
                'TolFun', 1e-8 ...
              );
j = 0;
figure;
for k=1:numel(exps)
    idx_exp = Data.Exp==exps(k); %selector for observations belonging to experiment exps(k)
    D = Data(idx_exp,:); %observations of experiment exps(k)
    yexp = D.In;%y_sim(idx_exp);%D.In;
    runs = unique(D.Run); %sweeps of that experiment
    CoeffNames = {'V_h','a','I_0','k'};
    subplot(3,3,k);
    hold on;
    for r=1:numel(runs)
        j = j + 1;
        fprintf('%i:\n', j);
        idx_r = D.Run==r; %selector for observations of sweep r
        Dr = D(idx_r,:);
        yr = yexp(idx_r);
        M{j,1} = exps(k); %experiment identifier
        M{j,2} = r; %sweep/run index
        M{j,3} = fitnlm( Dr.Vp, yr, ...
                         @inact_curve, ...
                         b0, ...
                         'CoefficientNames', CoeffNames, ...
                         'Options', opts ...
                       );
        Beta(j,:) = M{j,3}.Coefficients.Estimate;
        SE(j,:) = M{j,3}.Coefficients.SE;
        M{j,4} = Dr.Ato(1); %Ato concentration
        plot(Dr.Vp, yr,'.');
        plot(M{j,3}.Variables.X1,M{j,3}.Fitted,'-');
        fprintf('R^2=%g, p-value: %g\n',M{j,3}.Rsquared.Ordinary,coefTest(M{j,3}));
    end
    title(string(D.Exp(1)),'Interpreter','none');
end
```

```
1:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
```

```
R^2=0.99973, p-value: 1.02066e-36
2:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999392, p-value: 1.40254e-33
3:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999614, p-value: 1.82267e-35
4:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999472, p-value: 1.96144e-34
5:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.998887, p-value: 8.79282e-32
6:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999897, p-value: 2.24665e-40
7:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999583, p-value: 1.35655e-35
8:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999428, p-value: 1.36342e-34
9:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999002, p-value: 8.8849e-32
10:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.998863, p-value: 1.76539e-31
11:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999825, p-value: 3.0974e-38
12:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999918, p-value: 4.80363e-41
13:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999816, p-value: 2.37348e-38
14:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999812, p-value: 3.74692e-38
15:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999916, p-value: 3.94354e-41
16:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999899, p-value: 3.74065e-40
17:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999878, p-value: 1.33096e-39
18:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
```

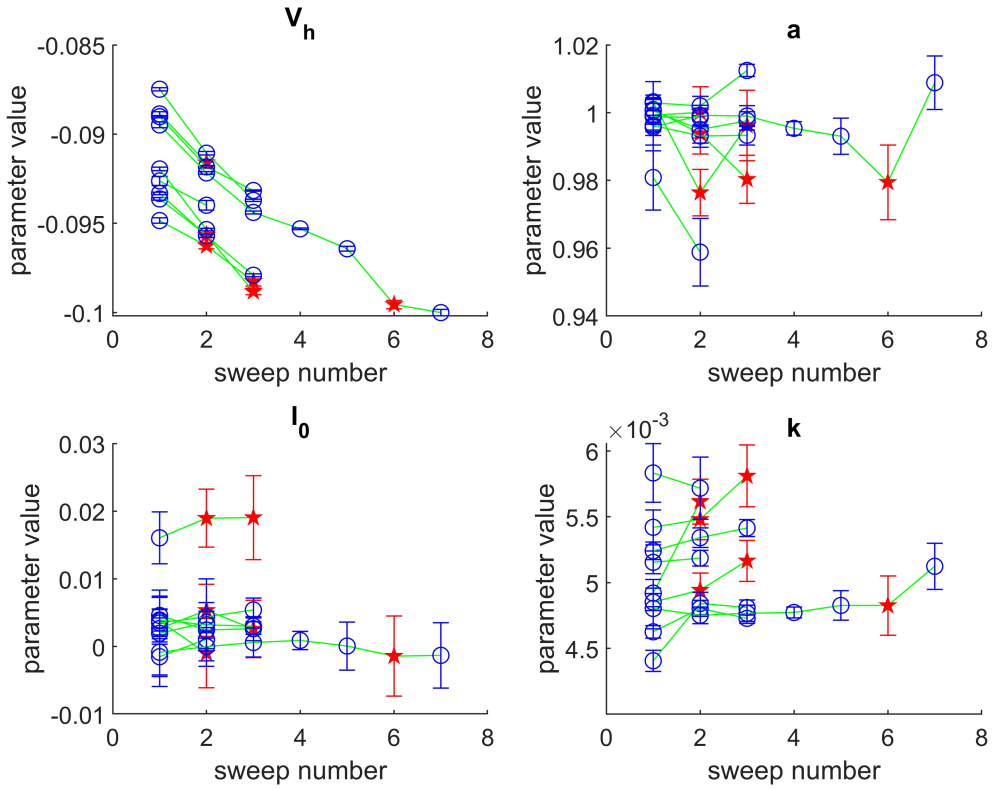
```
R^2=0.999908, p-value: 1.07132e-40
19:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999938, p-value: 4.56499e-42
20:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999905, p-value: 1.33241e-40
21:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999962, p-value: 6.01504e-44
22:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999795, p-value: 1.87809e-14
23:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999961, p-value: 1.19339e-16
24:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999957, p-value: 1.69128e-16
25:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999982, p-value: 1.1497e-17
26:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999873, p-value: 3.79694e-15
27:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999608, p-value: 9.22427e-12
28:
Iterations terminated: relative change in SSE less than OPTIONS.TolFun
R^2=0.999752, p-value: 2.42761e-14
```



Dependence of Parameter Values from Experiment and Sweep Number

Large part of variance between sweeps (runs) depends on the experiment for all parameters. Only for parameter V_h variance is also determined by sweep number.

```
figure;
for p=1:4
    subplot(2,2,p);
    hold on;
    for k=1:numel(exps)
        idx_exp = [M{:,1}]==exps(k);
        runs = [M{idx_exp,2}]; %sweep number
        ato = [M{idx_exp,4}]; %Ato concentration
        b = Beta(idx_exp, p); %parameter with index p
        se = SE(idx_exp, p);
        plot(runs,b,'-g');
        errorbar(runs(ato==0),b(ato==0),se(ato==0),'ob');
        errorbar(runs(ato>0),b(ato>0),se(ato>0),'pr','MarkerFaceColor','r');
    end
    title(CoeffNames{p});
    xlabel('sweep number');
    ylabel('parameter value');
end
```



Thus a random effects (hierarchical) nonlinear modelling approach will be applied. V_h will be expressed as an apparent value by linear dependence from sweep number

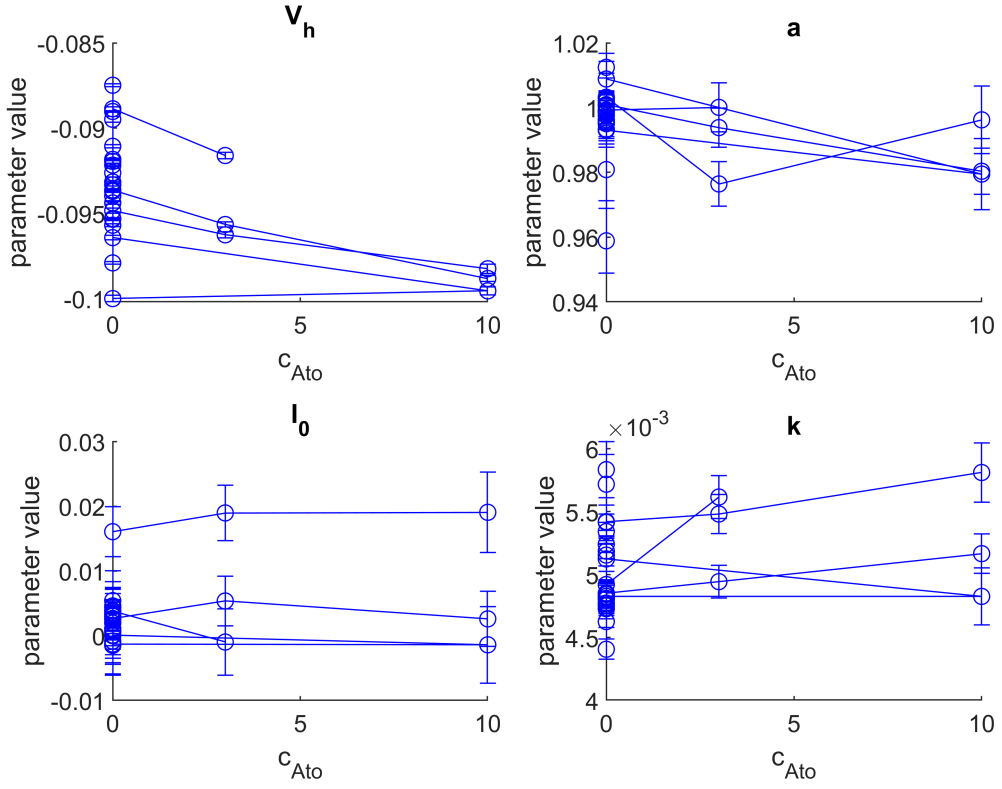
$$V_h^* = \hat{V}_h + \delta \cdot s$$

where δ represents the change of V_h per run.

Dependence of Parameter Values from Atomexitine Concentration

```
figure;
for p=1:4
    subplot(2,2,p);
    hold on;
    for k=1:numel(exps)
        idx_exp = [M{:,1}]==exps(k);
        c_ato = [M{idx_exp,4}]; %Ato concentration
        b = Beta(idx_exp,p);
        se = SE(idx_exp,p);
        errorbar(c_ato,b,se,'-ob');
    end
    title(CoeffNames{p});
    xlabel('c_{Ato}');
    ylabel('parameter value');

end
```



There might be a concentration dependence for V_h. There are only data at 2 concentrations available, not enough to fit a concentration-response curve. Therefore a constant shift parameter is introduced that takes effect only if $c_{\text{Ato}} > 0$, giving for V_h^*

$$V_h^* = \hat{V}_h + \delta_s \cdot s + \delta_{\text{Ato}}|_{c_{\text{Ato}}>0}$$

Problem Definition

$\hat{V}_h$, a, I_0, k get random components on the experiment level.

There are 3 predictors: The test potential V_p, the Atomexitine concentration c_{Ato} and the sweep number s. V_p is observation-specific, c_{Ato} and s are sweep-specific while no predictor is group-specific on the experiment level. It is assumed that there is no random influence on the sweep level so that the observation- and sweep-levels merge to one lower level, while the experiment introduces an upper level at which random effects on $\hat{V}_h, a, I_0$ and k take effect.

The fixed-effects parameter vector takes the form $\beta = \left[\hat{V}_h, a, I_0, k, \delta_s, \delta_{\text{Ato}}\right]$.

Fit the Nonlinear Mixed-Effects Model

```
b0 = [-.095, ... %1 half max inact. Vh
          1, ... %2 amplitude
```

```
          0, ... %3 residual current I0
       .005, ... %4 slope k
      -.0015,... %5 change of Vh/sweep
      -.003  ... %6 change of Vh by Ato
     ];
scale = [1e3 1 1e3 1e3 1e3 1e3];
b0 = b0.*scale;
X = Data{:,{'Run','Ato','Vp'}}; %predictors
y = Data.In; %response
grp = Data.Exp; %grouping on experiment level
CoeffNames = { 'V_h', 'a', 'I_0', 'k', ...
               'd_s', 'd_Ato' ...
             };
opts = statset('Display','final', ...
               'TolFun', 1e-8 ...
              );
[beta,PSI,stats,RE] = nlmefit( X,y,grp,[], ...
                               @(phi,X)modelfun(phi,X,scale), ...
                               b0, ...
                               'FEParamsSelect', 1:6, ... %all parameters have fixed effects
                               'REParamsSelect', [1 4], ... %first 4 parameters have random effe
                               'ApproximationType', 'RELME', ...
                               'Options', opts ...
                             );
```

```
      Beta_hat'   = [ -89.499102 0.994990 4.267955 5.177561 -1.980102 -0.454821 ]
      theta_hat'  = [ -9.937587 -6.797320 ]
      logLike     = 1287.486806
      sigma2_hat  = 0.000248
      Psi_hat     = [  5.138225  0.000000
                       0.000000  0.222337 ]
      Delta_hat   = [  0.006952  0.000000
                       0.000000  0.033418 ]
      Deg.Freedom = 9
      AIC         = -2556.973612
      BIC         = -2565.086101
 8 iterations and 2043 model evaluations in 4.921875 seconds.
```

What follows is a NHST for the Null-Hypothesis that a parameter value is zero. The test statistic (Wald-statistic) is computed from the standard error returned by nlmefit in the stats structure.

```
alpha = 0.01;
n = numel(y);
n = size(M,1);
b = beta(:);
b_se = stats.sebeta(:);
mu = zeros(size(b));
W = (b - mu)./b_se;
df = n - numel(b) - numel(RE) - 1;
p_value = 2*tcdf(-abs(W), df);
h = p_value<(alpha/numel(b));
Beta_Stats = table(b,b_se,W,p_value,h)
```

```
Beta_Stats = 6×5 table
```

	b	b_se	W	p_value	h
1	-89.4991	0.7612	-117.5721	0.0000	1
2	0.9950	0.0019	520.0341	0.0000	1
3	4.2680	1.2621	3.3816	0.0430	0
4	5.1776	0.1629	31.7837	0.0001	1
5	-1.9801	0.0426	-46.4491	0.0000	1
6	-0.4548	0.1468	-3.0982	0.0534	0

Model Comparison to Deduce Significance of Atomexitine Effect on Inactivation Shift

```
[beta0,PSI0,stats0,RE0] = nlmefit( X,y,grp,[], ...
                                   @(phi,X)modelfun_no_dato(phi,X,scale(1:5)), ...
                                   b0(1:5), ...
                                   'FEParamsSelect', 1:5, ... %all parameters have fixed effects
                                   'REParamsSelect', 1:4, ... %first 4 parameters have random effect
                                   'Options', opts ...
                                 );
```

```
    Beta_hat'   = [ -89.447292 0.994016 4.133197 5.153964 -2.040329 ]
    theta_hat'  = [ -9.977097 1.269347 -11.371545 -6.391624 ]
    logLike     = 1308.040515
    sigma2_hat  = 0.000226
    Psi_hat     = [  4.860531  0.000000  0.000000  0.000000
                     0.000000  0.000063  0.000000  0.000000
                     0.000000  0.000000 19.601293  0.000000
                     0.000000  0.000000  0.000000  0.134751 ]
    Delta_hat   = [  0.006816  0.000000  0.000000  0.000000
                     0.000000  1.886406  0.000000  0.000000
                     0.000000  0.000000  0.003394  0.000000
                     0.000000  0.000000  0.000000  0.040933 ]
    Deg.Freedom = 10
    AIC         = -2596.081029
    BIC         = -2594.108783
8 iterations and 2178 model evaluations in 9.953125 seconds.
```

```
[beta1,PSI1,stats1,RE1] = nlmefit( X,y,grp,[], ...
                                   @(phi,X)modelfun(phi,X,scale(1:6)), ...
                                   b0(1:6), ...
                                   'FEParamsSelect', 1:6, ... %all parameters have fixed effects
                                   'REParamsSelect', 1:4, ... %first 4 parameters have random effect
                                   'Options', opts ...
                                 );
```

```
    Beta_hat'   = [ -89.473185 0.994047 4.128934 5.154811 -1.979292 -0.459311 ]
    theta_hat'  = [ -9.946315 1.240312 -11.407336 -6.419010 ]
    logLike     = 1313.490402
    sigma2_hat  = 0.000221
    Psi_hat     = [  4.606062  0.000000  0.000000  0.000000
                     0.000000  0.000064  0.000000  0.000000
                     0.000000  0.000000 19.853759  0.000000
```

```
                    0.000000  0.000000  0.000000  0.135344 ]
      Delta_hat   = [  0.006921  0.000000  0.000000  0.000000
                       0.000000  1.859218  0.000000  0.000000
                       0.000000  0.000000  0.003334  0.000000
                       0.000000  0.000000  0.000000  0.040377 ]
      Deg.Freedom = 11
      AIC         = -2604.980804
      BIC         = -2602.811334
 9 iterations and 2277 model evaluations in 9.625000 seconds.
```

```
% llog1 = stats.logl;
% llog0 = stats0.logl;
% D = 2*(llog1 - llog0); %test statistic for LR Test
% df = numel(beta)+numel(RE)-numel(beta0)-numel(RE0);
% p_value = 1 - chi2cdf(D, df);
```

1. Fit model without and with δ_{Ato} (*reduced m*odel vs. *full* model).
2. Simulate data from the *full* model by *parametric sampling*.
3. Fit *reduced* and *full* models to simulated data.
4. Compare models with respect to *AIC*, *BIC* and *log-likelihood*.

```
% Marginal normal distributions of random effects
% [mu,sigma] = normfit(RE0');
[mu,sigma] = normfit(RE1');
% Simulate data
n_sim = 1000;

aic0 = zeros(n_sim,1);
bic0 = zeros(n_sim,1);
logl0 = zeros(n_sim,1);
aic1 = zeros(n_sim,1);
bic1 = zeros(n_sim,1);
logl1 = zeros(n_sim,1);
d_ato = zeros(n_sim,1);
y_sim = zeros(size(y));
opts = statset('Display','off', ...
               'TolFun', 1e-6 ...
              );

timer1 = tic; %timer
Warnings = false(n_sim,2);
run = 1;
Beta_sim = [];
for j=1:n_sim
    for k=1:numel(exps)
        idx_exp = Data.Exp==exps(k); %selector for an experiment
        re_exp = sigma.*randn(1,numel(sigma)) + mu; %random effects for this experiment
%         beta_exp = [ beta0(1:4)' + re_exp, ...
%                      beta0(5) ...
%                    ];
        beta_exp = [ beta1(1:4)' + re_exp, ...
                     beta1([5 6])' ...
```

```
                    ];
        X_exp = X(idx_exp,:); %predictors for experiment k
        n_sweeps = X_exp(end,1); %number of sweeps/runs in experiment k
        y = zeros(size(X_exp,1),1);
        for s=1:n_sweeps
%             eps_sim = stats0.sebeta.*randn(1,numel(stats0.sebeta));
            eps_sim = stats1.sebeta.*randn(1,numel(stats1.sebeta));
            beta_sim = beta_exp + eps_sim;
            Beta_sim(run,:) = beta_sim;
            run = run + 1;
            idx_sweep = X_exp(:,1)==s;
            yt = modelfun_no_dato(beta_sim(1:5),X_exp(idx_sweep,:),scale(1:5));
            y(idx_sweep) = yt + stats1.errorparam*randn(sum(idx_sweep),1);
        end
        y_sim(idx_exp) = y;
    end
    lastwarn(''); % set lastwarn() to empty
    [b_sim,~,stats_sim,~] = nlmefit( X,y_sim,grp,[], ...
                                @(phi,X)modelfun_no_dato(phi,X,scale(1:5)), ...
                                b0(1:5), ...
                                'FEParamsSelect', 1:5, ... %all parameters have fixed effects
                                'REParamsSelect', 1:4, ... %first 4 parameters have random effect
                                'ApproximationType', 'RELME', ...
                                'RefineBeta0', false, ...
                                'Options', opts ...
                               );
   Warnings(j,1) = ~isempty(lastwarn);
   aic0(j) = stats_sim.aic;
   bic0(j) = stats_sim.bic;
   logl0(j) = stats_sim.logl;
   lastwarn(''); % set lastwarn() to empty
   [b_sim1,~,stats_sim1,~] = nlmefit( X,y_sim,grp,[], ...
                                @(phi,X)modelfun(phi,X,scale(1:6)), ...
                                b0, ...
                                'FEParamsSelect', 1:6, ... %all parameters have fixed effects
                                'REParamsSelect', 1:4, ... %first 4 parameters have random effect
                                'ApproximationType', 'RELME', ...
                                'RefineBeta0', false, ...
                                'Options', opts ...
                               );
   Warnings(j,2) = ~isempty(lastwarn);
   aic1(j) = stats_sim1.aic;
   bic1(j) = stats_sim1.bic;
   logl1(j) = stats_sim1.logl;
   d_ato(j) = b_sim1(6);
   if any(Warnings(j,:))
     fprintf('%i: AIC0:%g|%i AIC1:%g|%i\n',j,aic0(j),Warnings(j,2),aic1(j),Warnings(j,2));
   end
end
```

```
25: AIC0:-2406.93|0 AIC1:-2414.59|0
Warning: Unable to find a step that will decrease SSE in the Gauss-Newton algorithm while solving PNLS.
Returning results from last iteration.
```

```
32: AIC0:-2302.37|1 AIC1:-2299.11|1
107: AIC0:-2406.25|0 AIC1:-2405.15|0
158: AIC0:-2242.48|1 AIC1:-2239.22|1
183: AIC0:-2329.2|1 AIC1:-2327.17|1
200: AIC0:-2319.94|0 AIC1:-2327.11|0
223: AIC0:-2413.9|1 AIC1:-2419.57|1
305: AIC0:-2364.37|0 AIC1:-2362.41|0
453: AIC0:-2406.75|1 AIC1:-2402.65|1
482: AIC0:-2380.05|0 AIC1:-2378.4|0
Warning: Unable to find a step that will decrease SSE in the Gauss-Newton algorithm while solving PNLS.
Returning results from last iteration.
522: AIC0:-2399.78|1 AIC1:-2432.27|1
Warning: Unable to find a step that will decrease SSE in the Gauss-Newton algorithm while solving PNLS.
Returning results from last iteration.
562: AIC0:-2519.96|0 AIC1:-2516.11|0
637: AIC0:-2419.4|1 AIC1:-2415.74|1
773: AIC0:-2458.69|1 AIC1:-2459.09|1
Warning: Unable to find a step that will decrease SSE in the Gauss-Newton algorithm while solving PNLS.
Returning results from last iteration.
780: AIC0:-2375.93|1 AIC1:-2372.35|1
845: AIC0:-2301.75|0 AIC1:-2299.3|0
Warning: Unable to find a step that will decrease SSE in the Gauss-Newton algorithm while solving PNLS.
Returning results from last iteration.
874: AIC0:-2429.59|1 AIC1:-2426.26|1
909: AIC0:-2408.26|0 AIC1:-2415.39|0
929: AIC0:-2240.82|1 AIC1:-2254.29|1
Warning: Unable to find a step that will decrease SSE in the Gauss-Newton algorithm while solving PNLS.
Returning results from last iteration.
930: AIC0:-2384.58|1 AIC1:-2381.26|1
```

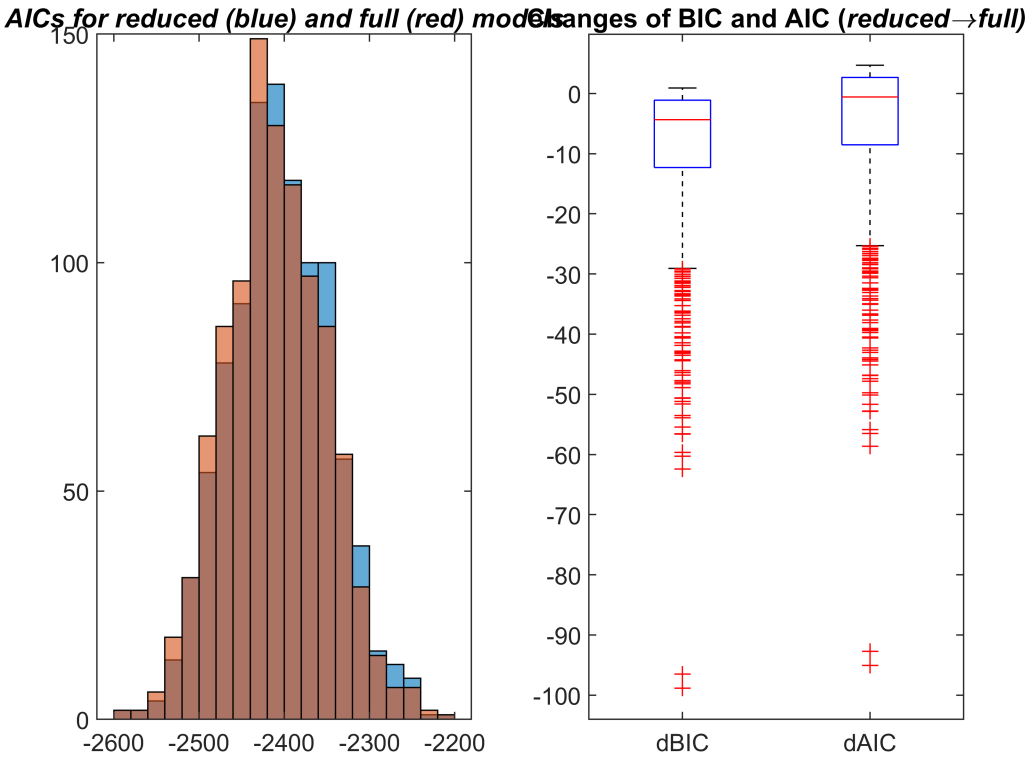
```
fprintf('total time taken: %g s\n',toc(timer1));
```

```
total time taken: 1181.3 s
```

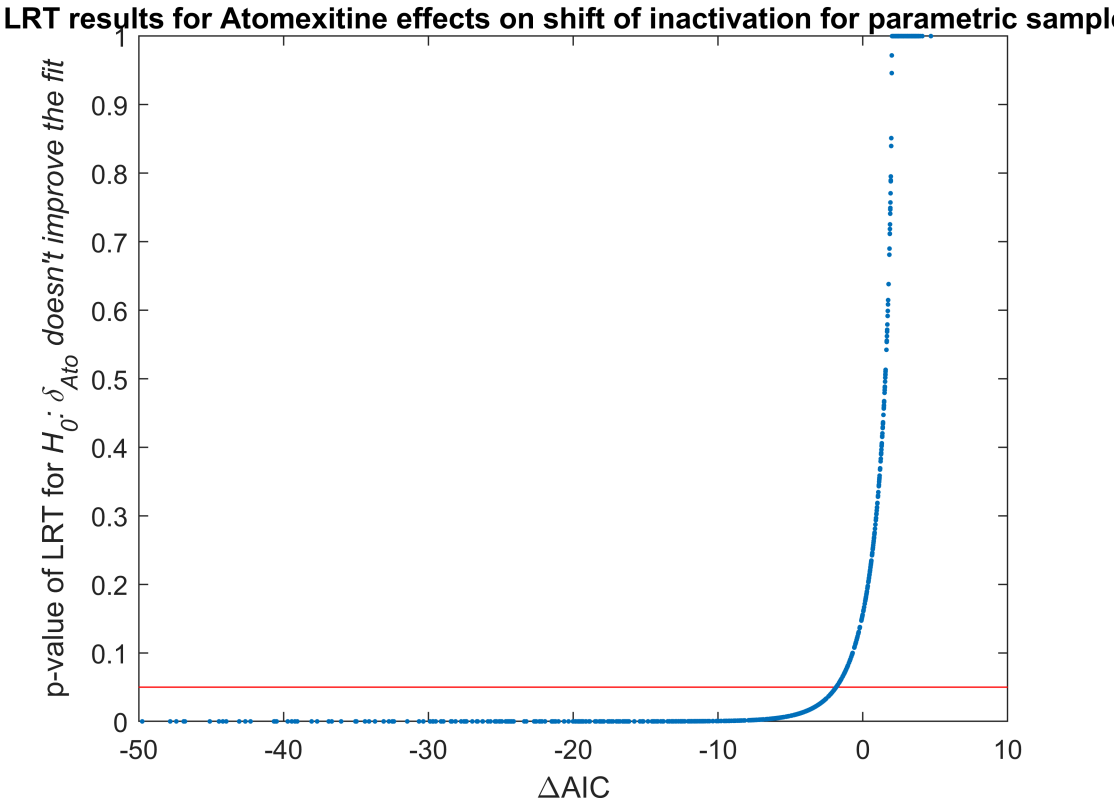
```
dAIC = aic1 - aic0; %Change of Akaike's Information Criterion
dBIC = bic1 - bic0;
dlogl = logl1 - logl0;
fprintf('max|95%%|50%%|5%%|min dAIC: %g|%g|%g|%g|%g\n',max(dAIC),prctile(dAIC,95),prctile(dAIC,
```

```
max|95%|50%|5%|min dAIC: 4.70004|3.64735|-0.568652|-29.6809|-95.09
```

```
figure;
subplot(1,2,1);
histogram(aic0);
hold on
histogram(aic1);
title('\it{AICs} for \it{reduced} (blue) and \it{full} (red) models')
subplot(1,2,2);
boxplot([dBIC dAIC],'Labels',{'dBIC','dAIC'});
title('Changes of BIC and AIC (\it{reduced}\rightarrow\it{full})')
```




```
pval = 1 - chi2cdf(2*dlogl,1);
figure;
plot([min(dAIC) 50],[.05 .05],'r-');
hold on;
plot(dAIC,pval,'.');
xlabel('\DeltaAIC');
ylabel('p-value of LRT for \it{H_0}: \delta_{Ato} doesn''t improve the fit');
xlim([-50 10]);
title('LRT results for Atomexitine effects on shift of inactivation for parametric samples')
```



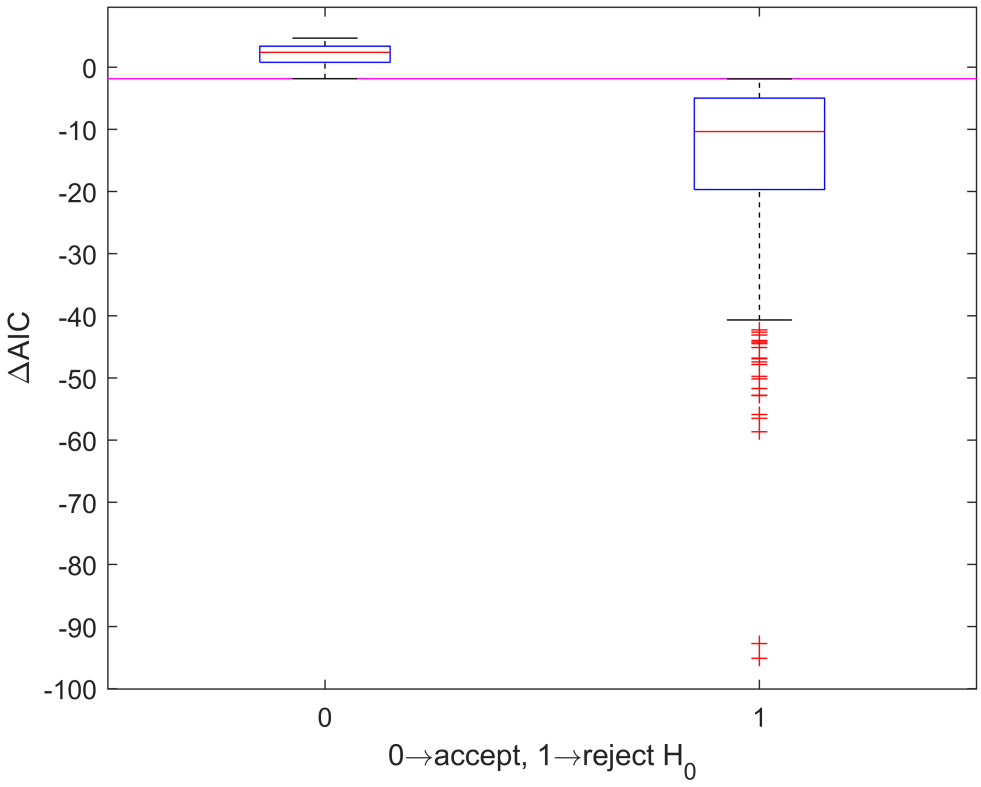

```
smallest_dAIC = min(dAIC(pval>=0.05));
fprintf('Rejection of H0 if dAIC<%g.\n',smallest_dAIC);
```

```
Rejection of H0 if dAIC<-1.83395.
```

```
fprintf('Fraction of parametric samples resulting in a p-value>=0.05 in Likelihood-Ratio testi
```

```
Fraction of parametric samples resulting in a p-value>=0.05 in Likelihood-Ratio testing: 0.549.
```

```
figure;
plot([-.5 3],[smallest_dAIC smallest_dAIC],'m-');
hold on;
boxplot(dAIC,pval<0.05);
ylabel('\DeltaAIC');
xlabel('0\rightarrowaccept, 1\rightarrowreject H_0');
```



Local Function Definitions

```
function y = inact_curve(b,vp)
  vh = b(1);
  a = b(2);
  i0 = b(3);
  k = b(4);
  dv = vp - vh;
  y = i0 + a./(1+exp(dv/k));
end

function y = modelfun(phi,X,scale)
% phi is the parameter vector
%   phi(1): Vh
%   phi(2): a
%   phi(3): I0
%   phi(4): k
%   phi(5): ds
%   phi(6): dato
% X is a n-by-3 matrix of predictors for each of n observations
%   X(:,1) give the run/sweep number
%   X(:,2) give the atomexitine concentration
phi = phi./scale;
sweeps = unique(X(:,1)); %sweeps
n_runs = numel(sweeps); %number of sweeps
n_steps = sum(X(:,1)==1);
```

```
Y = NaN(n_steps,n_runs);
for s=1:n_runs
   sel = X(:,1)==sweeps(s); %selector for sweep s
   c_ato = X(sel,2); %atomexitine concentrations during sweep (constant)
   vp = X(sel,3); %test potentials for each observation in sweep s
   vh_app = phi(1) + phi(5)*sweeps(s) + phi(6)*(c_ato(1)>0);  %half max. inactivation
   dv = vp - vh_app;
   Y(1:numel(vp),s) = phi(3) + phi(2)./(1+exp(dv/phi(4)));
end
y = Y(:);
y = y(~isnan(y));
end

function y = modelfun_no_dato(phi,X,scale)
% phi is the parameter vector
%   phi(1): Vh
%   phi(2): a
%   phi(3): I0
%   phi(4): k
%   phi(5): ds
% X is a n-by-3 matrix of predictors for each of n observations
%   X(:,1) give the run/sweep number
%   X(:,2) give the atomexitine concentration
phi = phi./scale;
sweeps = unique(X(:,1)); %sweeps
n_runs = numel(sweeps); %number of sweeps
n_steps = sum(X(:,1)==1);
Y = NaN(n_steps,n_runs);
for s=1:n_runs
   sel = X(:,1)==sweeps(s); %selector for sweep s
%    c_ato = X(sel,2); %atomexitine concentrations during sweep (constant)
   vp = X(sel,3); %test potentials for each observation in sweep s
   vh_app = phi(1) + phi(5)*sweeps(s);  %half max. inactivation
   dv = vp - vh_app;
   Y(1:numel(vp),s) = phi(3) + phi(2)./(1+exp(dv/phi(4)));
end
y = Y(:);
y = y(~isnan(y));
end
```