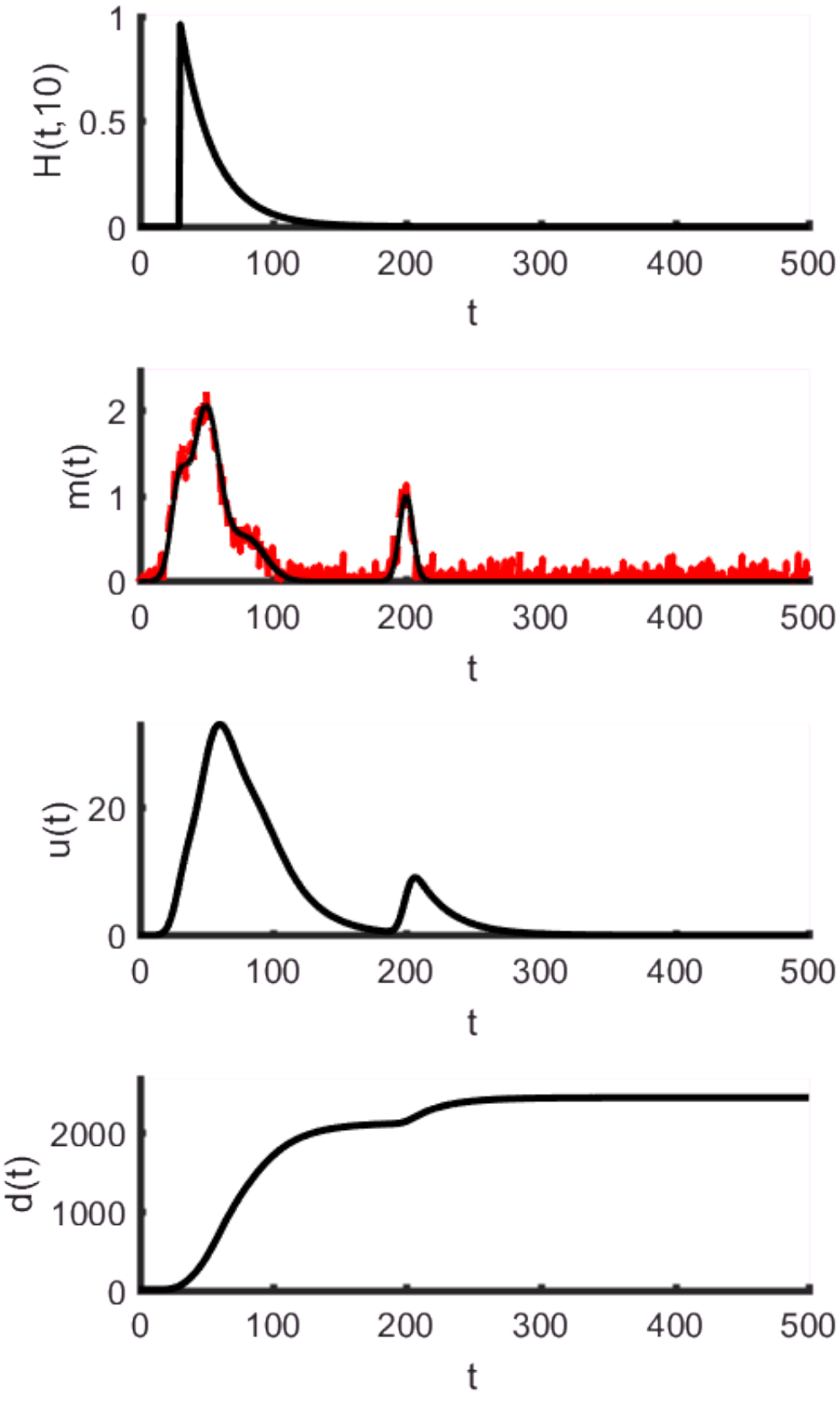
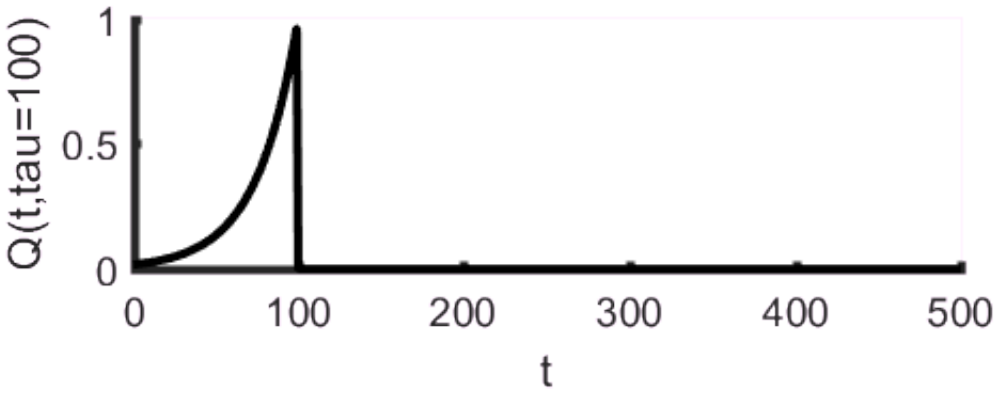
```
% gda11_06
% Supports Figures 11.8 and 11.9.
% temperature / chemical reaction example
% adjoint method used to build data kernel
% and the problem is then solved usign least squares

% kernel for differential equation
clear all;

% time variable t
M=501;
N=M;
Dt=1;
t = Dt*[0:M-1]';
tmax=max(t);

% differential equation (d/dt + c)u = f
% has green function G(t,t')=H(t-t')exp(-c(t-t'))
c = 0.04;

% Forward problem

% create a heat function m
% allow two options, complex and simple
if( 1 )
    % complicated m built out of Gaussians
    t1 = 30;
    sigma1 = 6;
    m = exp(-0.5*((t-t1).^2)/(sigma1^2));
    t1 = 50;
    sigma1 = 10;
    m = m+2*exp(-0.5*((t-t1).^2)/(sigma1^2));
    t1 = 80;
    sigma1 = 14;
    m = m+0.5*exp(-0.5*((t-t1).^2)/(sigma1^2));
    t1 = 200;
    sigma1 = 5;
    m = m+exp(-0.5*((t-t1).^2)/(sigma1^2));
else
    % very simple m for test purposes
    m=zeros(M,1);
    m(floor(1*M/4))=1;
    m(floor(2*M/4))=0.5;
end

% perform the Green function integral to predict u
% use convolution function; faster than coding the integral
Ho = exp(-c*t);
u = Dt*conv(Ho,m); % Green function integral is a convolution
u=u(1:M);

% integrate u to get P
b=1;
P = Dt*b*cumsum(u);

% data is P with random noise
dtrue = P;
sigmad = 0.1;
dobs = dtrue+random('Normal',0,sigmad,N,1);
```

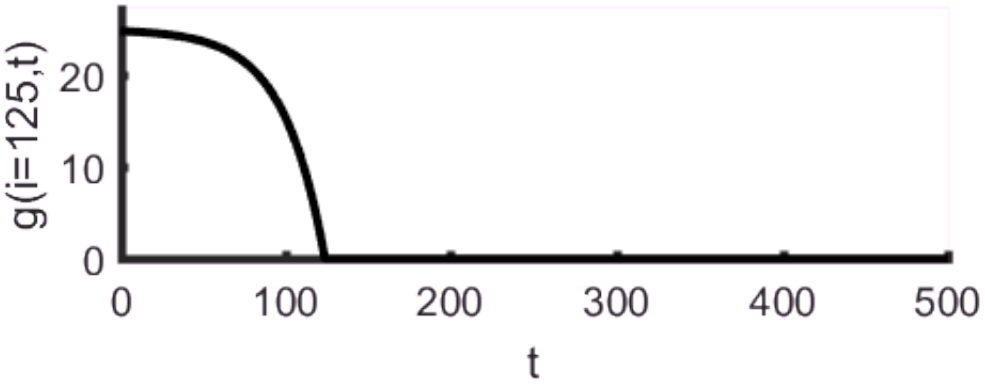
```
% inverse problem

% adjoint differential equation (-d/dt + c)u = h
% has green function G(t,t')=H(t'-t)exp(c(t-t'))

% data kernel Gd
N=M;
Gd = zeros(N,M);
for i = [1:N]
for j = [1:M]
    ti = i*Dt;
    tee = j*Dt;
    if( tee <= ti )
        Gd(i,j)=Dt*(-b/c)*(exp(-c*(ti-tee))-1);
    else
        Gd(i,j)=0;
    end
end
end



% solve with dampled least squares
e2 = 1.0e-1;
mest = (Gd'*Gd + e2*eye(M,M))\(Gd'*dobs);


% time series plots
figure(2);
figure('pos',[10 10 500 900]);
clf;

% plot Green function for t'=10;
subplot(4,1,1);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
axis( [0, tmax, 0, 1] );
tp=30;
H = (t>tp).*exp(-c*(t-tp));
plot( t, H, 'k-', 'LineWidth', 3 );
xlabel('t');
ylabel('H(t,10)');

% plot heat function
subplot(4,1,2);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
axis( [0, tmax, 0, 1.2*max(m)] );
plot( t, mest, 'r--', 'LineWidth', 3 );
plot( t, m, 'k-', 'LineWidth', 2 );
xlabel('t');
ylabel('m(t)');

% perform the Green function integral to predict u
% use convolution function; faster than coding the integral
subplot(4,1,3);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
```

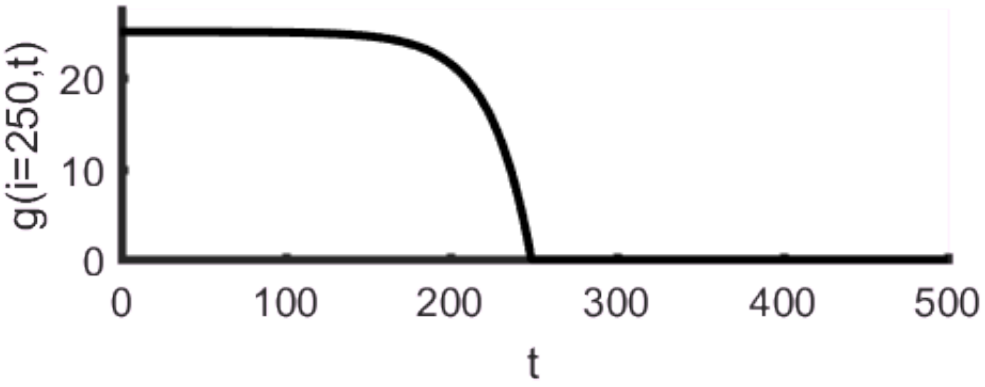
```
hold on;
axis( [0, tmax, 0, max(u)] );
plot( t, u, 'k-', 'LineWidth', 3 );
xlabel('t');
ylabel('u(t)');


% plot the observed data
subplot(4,1,4);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
axis( [0, tmax, 0, 1.1*max(dobs)] );
plot( t, dobs, 'k-', 'LineWidth', 3 );
xlabel('t');
ylabel('d(t)');
```

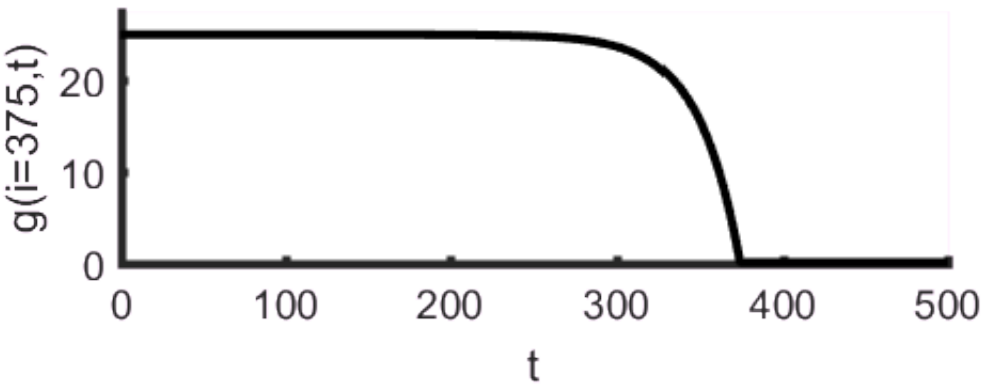

H(t,10)
1
0.5
0
0
100
200
300
400
500
t
m(t)
2
1
0
0
100
200
300
400
500
t
u(t)
20
0
0
100
200
300
400
500
t
d(t)
2000
1000
0
0
100
200
300
400
500
t

```
% Figure 11.8 Example of the solution of a continuous inverse problem involving a differential
% equation. (A) Green function F(t,τ) for τ = 30. (B) True (black) and estimated (red) heat
% production function m(t). (C) Temperature u(t), which solves ℒu = m. (D) Observed data d(t),
% which is proportional to the integral of u(t). MatLab script gda11_06.

figure(3);
clf;
figure('pos',[10 10 500 900]);

subplot(4,1,1);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
axis( [0, tmax, 0, 1] );
tp=100;
H = Dt*(t<tp).*exp(c*(t-tp));
plot( t, H, 'k-', 'LineWidth', 3 );
xlabel('t');
ylabel('Q(t,tau=100)');

% plot a few rowa of the data kernel
subplot(4,1,2);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
i=floor(1*N/4);
axis( [0, tmax, 0, 1.1*max(Gd(i,:)) ] );
plot( t, Gd(i,:)', 'k-', 'LineWidth', 3 );
xlabel('t');
ylabel('g(i=125,t)');

subplot(4,1,3);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
i=floor(2*N/4);
axis( [0, tmax, 0, 1.1*max(Gd(i,:)) ] );
plot( t, Gd(i,:)', 'k-', 'LineWidth', 3 );
xlabel('t');
ylabel('g(i=250,t)');

subplot(4,1,4);
set(gca,'LineWidth',3);
set(gca,'FontSize',14);
hold on;
i=floor(3*N/4);
axis( [0, tmax, 0, 1.1*max(Gd(i,:)) ] );
plot( t, Gd(i,:)', 'k-', 'LineWidth', 3 );
xlabel('t');
ylabel('g(i=375,t)');
```


Q(t,tau=100)
1
0.5
0
0
100
200
300
400
500
t


g(i=125,t)
20
10
0
0
100
200
300
400
500
t


g(i=250,t)
20
10
0
0
100
200
300
400
500
t


g(i=375,t)
20
10
0
0
100
200
300
400
500
t

```
% Figure.  (A) Adjoint Green Function fot tau=100.  (B-D) Data kernel
% g(i, t) for t=125, 250, ane 375.

% draw the data kernel
figure('pos',[10 10 500 500]);
gda_draw(Gd);
```