Chapter 9

Answers Chapter 9

Exercises

9.1.4

a) Solving the equations for each year yields 5 slightly different values for a, ranging from 0.0148 to 0.0169. This range is used as the initial edges for the bisection method. The result is $a = 0.0165$, see program **ex914.m**.

b) See Figure 9.1.

c) The calculated population in 2000 is 5.7722 billion, the actual population was 6.0849. (Numbers taken from U.S. Census Bureau.) The error is 5.14%.

d) Newton's method results in $a = 0.0165$ after three iterations. See program **ex914.m**.

9.1.5

a) Using Newton's method results in $r = 2.5512, a = 0.0169$, after two iterations. See program **ex915.m**

b) The calculated population in 2000 is 5.9271 billion, the actual population was 6.0849. The error is 2.59%.

9.1.6

a)

$$\frac{\partial J}{\partial a} = \sum_{t=0}^{t=10} \left(r(t; a, r_0, R) - d_t\right) \frac{r_0 R t e^{-at}(R - r_0)}{\left(r_0 + e^{-at}(R - r_0)\right)^2}$$

$$\frac{\partial J}{\partial r_0} = \sum_{t=0}^{t=10} \left(r(t; a, r_0, R) - d_t\right) \frac{R^2 e^{-at}}{\left(r_0 + e^{-at}(R - r_0)\right)^2}$$

$$\frac{\partial J}{\partial R} = \sum_{t=0}^{t=10} \left(r(t; a, r_0, R) - d_t\right) \frac{r_0^2 \left(1 - e^{-at}\right)}{\left(r_0 + e^{-at}(R - r_0)\right)^2}$$

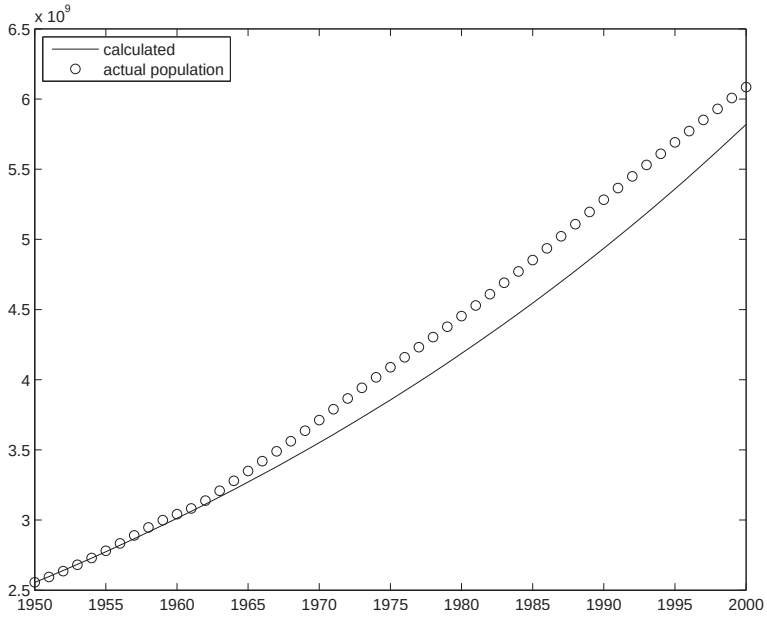

Figure 9.1: Excercise 9.1.4 b) World Population.

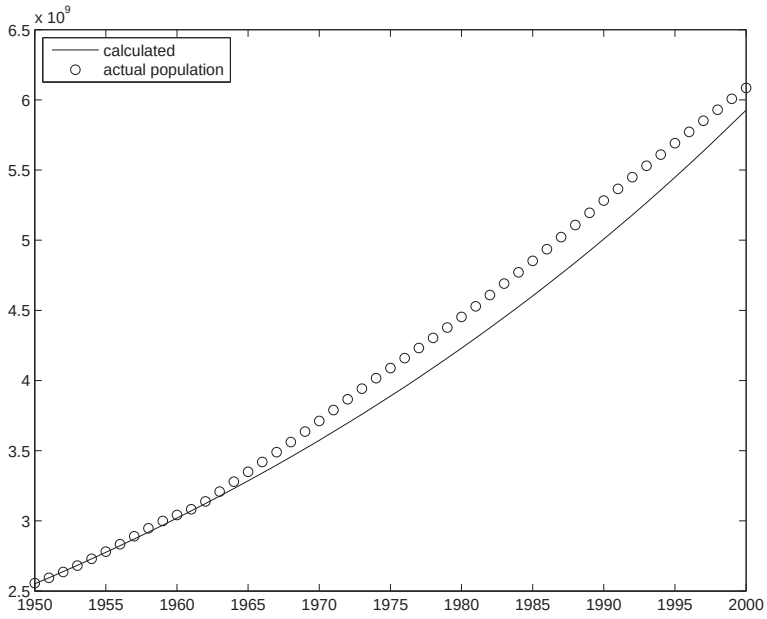

Figure 9.2: Excercise 9.1.5 b) World Population.

b) For

$$a = 0,$$
$$r_0 = R = \frac{1}{11}\sum_{t=0}^{t=10} d_t,$$

$$\frac{\partial J}{\partial a} = \sum_{t=0}^{t=10} \left(r(t;a,r_0,R) - d_t\right) \frac{r_0 R t (R - r_0)}{\left(r_0 + (R - r_0)\right)^2} = \sum_{t=0}^{t=10} \left(r(t;a,r_0,R) - d_t\right) \frac{0}{\left(r_0 + (R - r_0)\right)^2} = 0.$$

When $a = 0$, we have that $r'(t) = 0$, and $r(t) = r(0) = r_0$.

$$\frac{\partial J}{\partial r_0} = \sum_{t=0}^{t=10} \left(r(t;a,r_0,R) - d_t\right) \frac{R^2}{\left(r_0 + (R - r_0)\right)^2} = \sum_{t=0}^{t=10} \left(r(t;a,r_0,R) - d_t\right) = \sum_{t=0}^{t=10} r_0 - \sum_{t=0}^{t=10} d_t = 0$$

$$\frac{\partial J}{\partial R} = \sum_{t=0}^{t=10} \left(r(t;a,r_0,R) - d_t\right) \frac{r_0^2 (1 - 1)}{\left(r_0 + (R - r_0)\right)^2} = \sum_{t=0}^{t=10} \left(r(t;a,r_0,R) - d_t\right) \frac{0}{\left(r_0 + (R - r_0)\right)^2} = 0$$

c) See program **ex916.m**

d) Newton's method returns the starting position.

e) for

$$(\widetilde{a}, \widetilde{r_0}, \widetilde{R}) = (0.05, 5.0, 11.0),$$

Newton's method returns

$$(0.036, 5.285, 9.213).$$

f) for

$$(\widetilde{a}, \widetilde{r_0}, \widetilde{R}) = (0.01, 5.0, 11.0),$$

Newton's method returns

$$(0.0, r = 5.687, R = 5.687).$$

g) for

$$(\widetilde{a}, \widetilde{r_0}, \widetilde{R}) = (0.8, 5.0, 11.0),$$

Newton's method returns

$$(-19.23, r = 5.284, R = 8.526).$$

h) In section 5.5.2 we find $R = 11.44$, and $a = 0.027$. The initial guess is important.

Programs (MATLAB)

ex914.m

```
function ex914
%
% Bisection and Newton's method for finding 'a' in population growth
% Starting points: 0.0148 and 0.0169 for bisection,
% 0.0148 for Newton

% Bisection;
ls =   0.0148;
rs =   0.0169;

while ((rs-ls) > 0.00001)
  midd = (ls+rs)*0.5;
  if ( ex914a(ls)*ex914a(midd) < 0)
    rs=midd;
  else
    ls = midd;
  end
 end
Bres = midd

% Newton:
x = 0.0148;
d =100;
j=0;
while (d > 0.00001)
 xp = x - ex914a(x)/ex914d(x);
 d= abs(x-xp);
 x=xp;
 j=j+1
 end

Nres = x

% Create figure to compare with actual numbers:

a=Nres;
t=[0:50];
y=t+1950;
Pop=2.555*1e9*exp(t*a);

USC=[2555948654,2593751978,2635830950,2681188031,2729296850,
2780907497,2834042890,2890057665,2946761863,2999303775,
3041593413,3082587875,3138805626,3208557765,3279921533,
3349157095,3419597944,3489539835,3561584085,3636415012,
3711800786,3789459117,3865863923,3941725131,4016384740,
4089026773,4160182844,4231882118,4303414792,4377673820,
```

```
4452766336,4528969006,4609080226,4690922711,4771126065,
4852574467,4935986347,5021959865,5108492222,5194873063,
5282371928,5365708797,5448725522,5529987425,5610065463,
5690982026,5771360981, 5850822196,5929679830,6007529585,
6084907596];

plot(y,Pop,y,USC,'bo')
legend('calculated','actual population','Location','NorthWest')

function  Gs = ex914a(a)
  dt = [2.555, 2.595, 2.635, 2.680, 2.728, 2.780]*1e9;
  t = [1, 2, 3, 4, 5];
  G= (2.555e9*exp(a*t) - dt(2:6)).*(2.555e9*t.*exp(a*t));
  Gs = sum(G);


function  dGs = ex914d(a)
   dt = [2.555, 2.595, 2.635, 2.680, 2.728, 2.780]*1e9;
   t = [1, 2, 3, 4, 5];
   dG=2.555e9*2.555e9*2*t.*t.*exp(2*a*t) -2.555e9*dt(2:6).*t.*t.*exp(a*t);
   dGs=sum(dG);
```

ex915.m

```
function ex915
%
% Newton's method for finding 'a' and 'r0' in population growth
% Starting Points: a = 0.0165, r0 = 2.555 billion

global dt;
global t;
 dt = [2.555, 2.595, 2.635, 2.680, 2.728, 2.780];
 t = [0, 1, 2, 3, 4, 5];

r = 2.555;
a = 0.0165;

it = 0;
C2 = 42.0;

% Newton:
x = [r;a]
while (C2 > 1.0e-6)
  M = [dFdr(x(1),x(2)), dFda(x(1),x(2)); dGdr(x(1),x(2)), dGda(x(1),x(2)) ];
  V = [F(x(1),x(2));G(x(1),x(2))];
  xn = x-inv(M)*V;
  x=xn
  C2 = abs(F(x(1),x(2))) + abs(G(x(1),x(2)))
```

```
  it = it+1
end

r=x(1)
a=x(2)

% Create figure to compare with actual numbers:

t=[0:50];
y=t+1950;
Pop=r*1e9*exp(t*a);

USC=[2555948654,2593751978,2635830950,2681188031, 2729296850,
2780907497,2834042890,2890057665,2946761863,2999303775,
3041593413,3082587875,3138805626,3208557765,3279921533,
3349157095,3419597944,3489539835,3561584085,3636415012,
3711800786,3789459117,3865863923,3941725131,4016384740,
4089026773,4160182844,4231882118,4303414792,4377673820,
4452766336,4528969006,4609080226,4690922711,4771126065,
4852574467,4935986347,5021959865,5108492222,5194873063,
5282371928,5365708797,5448725522,5529987425,5610065463,
5690982026,5771360981, 5850822196,5929679830,6007529585,
6084907596];

plot(y,Pop,y,USC,'bo')
legend('calculated','actual population','Location','NorthWest')

function f = F(r,a)
  global t;
  global dt;
  v = (r*exp(a*t) -dt)*r.*t.*exp(a*t);
  f = sum(v);

function g = G(r,a)
  global t;
  global dt;
  v = (r*exp(a*t) -dt).*exp(a*t);
  g = sum(v);

function dFdr = dFdr(r,a)
  global t;
  global dt;
  v = 2*r*t.*exp(2*a*t) -dt.*t.*exp(a*t);
  dFdr= sum(v);

function dFda = dFda(r,a)
   global t;
  global dt;
 v = 2*r*r*t.*t.*exp(2*a*t) -dt*r.*t.*t.*exp(a*t);
  dFda= sum(v);
```

```
function dGdr = dGdr(r,a)
  global t;
  global dt;
  v = exp(2*a*t);
  dGdr= sum(v);

function dGda = dGda(r,a)
  global t;
  global dt;
  v = r*2*t.*exp(2*a*t)-t.*dt.*exp(a*t);
  dGda= sum(v);
```

ex916.m

```
function ex916
%
%  Solves the expression for 'J' using Newton's method

global dt;
global t;
 dt = [5.284, 5.367, 5.450, 5.531, 5.611, 5.691, 5.769, 5.847, 5.925, 6.003, ...
       6.080];
 t = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

% NEWTON Starting point: a=0.025 r=5.0 R=11.0
a = 0.05;
r = 5.0;
R= 11.0;

it=0;
C2 = 300.0;

while (C2 > 1.0e-6)
 x = [a;r;R];
 M = [d2Jdada(a,r,R),d2Jdadr(a,r,R),d2JdadR(a,r,R);d2Jdadr(a,r,R),
      d2Jdrdr(a,r,R),d2JdrdR(a,r,R);d2JdadR(a,r,R),d2JdrdR(a,r,R),
      d2JdRdR(a,r,R)  ];
 V = [dJda(a,r,R);dJdr(a,r,R);dJdR(a,r,R)];
 xn = x-inv(M)*V;
  it = it+1;
 a = xn(1);
 r = xn(2);
 R = xn(3);
 C2 = abs(dJda(a,r,R)) + abs(dJdr(a,r,R))+ abs(dJdR(a,r,R));
end

% functions used
function J = J(a,r,R)
  global t;
```

```
  global dt;
      v =  (r*R./(r + exp(-a*t)*(R-r))) - dt;
      J = 0.5*sum(v.*v);


function dJda = dJda(a,r,R)
   global t;
   global dt;
      Funcr  = R*r./(r+exp(-a*t)*(R-r));
      f = r+exp(-a*t)*(R-r);
      v= (Funcr-dt)*R*r.*t.*exp(-a*t)*(R-r)./(f.*f);
      dJda = sum(v);


function d2Jdada = d2Jdada(a,r,R)
  global t;
  global dt;
       Funcr  = R*r./(r+exp(-a*t)*(R-r));
       f = r+exp(-a*t)*(R-r);
       drda = R*r.*t.*exp(-a*t)*(R-r)./(f.*f);
       drda2 = R*r*(R-r)*t.*t.*exp(-a*t).*(exp(-a*t)*(R-r)-r)./(f.*f.*f);
       tot = drda.*drda + (Funcr-dt).*drda2;
      d2Jdada = sum(tot);


function d2Jdadr = d2Jdadr(a,r,R)
  global t;
  global dt;
       Funcr  = R*r./(r+exp(-a*t)*(R-r));
       f = r+exp(-a*t)*(R-r);
       drda = R*r.*t.*exp(-a*t)*(R-r)./(f.*f);
       drdr = R*R*exp(-a*t)./(f.*f);
       drdrda = R*R*t.*exp(-a*t).*((R-r)*exp(-a*t) -r)./(f.*f.*f);
       tot = drdr.*drda + (Funcr-dt).*drdrda;
       d2Jdadr = sum(tot);


function d2JdadR = d2JdadR(a,r,R)
  global t;
  global dt;
       Funcr  = R*r./(r+exp(-a*t)*(R-r));
       f = r+exp(-a*t)*(R-r);
       drda = R*r.*t.*exp(-a*t)*(R-r)./(f.*f);
       drdR = r*r.*(1-exp(-a*t))./(f.*f);
       drdRda = r*r*t.*exp(-a*t).*(-exp(-a*t)*(R-r) +2*R -r)./(f.*f.*f);
       tot = drdR.*drda + (Funcr-dt).*drdRda;
       d2JdadR = sum(tot);


function d2Jdrdr = d2Jdrdr(a,r,R)
```

```
  global t;
  global dt;
       Funcr  = R*r./(r+exp(-a*t)*(R-r));
       f = r+exp(-a*t)*(R-r);
       drdr =  R*R*exp(-a*t)./(f.*f);
       drdr2 = 2*R*R*exp(-a*t).*(exp(-a*t)-1)./(f.*f.*f);
       tot = drdr.*drdr + (Funcr-dt).*drdr2;
      d2Jdrdr = sum(tot);


function d2JdrdR = d2JdrdR(a,r,R)
  global t;
  global dt;
       Funcr  = R*r./(r+exp(-a*t)*(R-r));
       f = r+exp(-a*t)*(R-r);
       drdr =  R*R*exp(-a*t)./(f.*f);
       drdR = r*r.*(1-exp(-a*t))./(f.*f);
       drdrdR = 2*R*r*exp(-a*t).*(1-exp(-a*t))./(f.*f.*f);
         tot = drdr.*drdR + (Funcr-dt).*drdrdR;
       d2JdrdR = sum(tot);

function d2JdRdR = d2JdRdR(a,r,R)
  global t;
  global dt;
       Funcr  = R*r./(r+exp(-a*t)*(R-r));
       f = r+exp(-a*t)*(R-r);
       drdR = r*r.*(1-exp(-a*t))./(f.*f);
       drdRdR = 2*r*r*exp(-a*t).*(exp(-a*t)-1)./(f.*f.*f);
       tot = drdR.*drdR + (Funcr-dt).*drdRdR;
      d2JdRdR = sum(tot);


function dJdr = dJdr(a,r,R)
   global t;
   global dt;
      Funcr  = R*r./(r+exp(-a*t)*(R-r));
      f = r+exp(-a*t)*(R-r);
      v= (Funcr-dt)*R*R.*exp(-a*t)./(f.*f);
      dJdr = sum(v);


function dJdR = dJdR(a,r,R)
   global t;
   global dt;
      Funcr  = R*r./(r+exp(-a*t)*(R-r));
      f = r+exp(-a*t)*(R-r);
      v= (Funcr-dt)*r*r.*(1-exp(-a*t))./(f.*f);
      dJdR = sum(v);
```

(Figure may be created using similar code as in 9.1.4 – 9.1.5.)