Exercise 12

Q.1. Observed: x_c, x_o, x_s, x_n factors: f_v, f_p, f_n

$$x_c \downarrow x_1,\quad x_o \downarrow x_2,\quad x_s \downarrow x_3,\quad x_n \downarrow x_4$$

(a). Factor analysis model:

$$\begin{aligned}
x_{i,1} &= \lambda_{11} f_{i1} + \lambda_{12} f_{i2} + \lambda_{13} f_{i3} + \epsilon_{i,1} + \mu_c \\
x_{i,2} &= \lambda_{21} f_{i1} + \lambda_{22} f_{i2} + \lambda_{23} f_{i3} + \epsilon_{i,2} + \mu_o \\
x_{i,3} &= \lambda_{31} f_{i1} + \lambda_{32} f_{i2} + \lambda_{33} f_{i3} + \epsilon_{i3} + \mu_s \\
x_{i,4} &= \lambda_{41} f_{i1} + \lambda_{42} f_{i2} + \lambda_{43} f_{i3} + \epsilon_{i,4} + \mu_n
\end{aligned}$$

(b). Matrix Notation for factor analysis model:

$$\begin{bmatrix} x_{i,1} \\ x_{i,2} \\ x_{i,3} \\ x_{i,4} \end{bmatrix} = \begin{bmatrix} \lambda_{11} & \lambda_{12} & \lambda_{13} \\ \lambda_{21} & \lambda_{22} & \lambda_{23} \\ \lambda_{31} & \lambda_{32} & \lambda_{33} \\ \lambda_{41} & \lambda_{42} & \lambda_{43} \end{bmatrix} \begin{bmatrix} f_{i,1} \\ f_{i,2} \\ f_{i,3} \end{bmatrix} + \begin{bmatrix} \epsilon_{i,1} \\ \epsilon_{i,2} \\ \epsilon_{i,3} \\ \epsilon_{i,4} \end{bmatrix} + \begin{bmatrix} \mu_c \\ \mu_o \\ \mu_3 \\ \mu_n \end{bmatrix}$$

(c). Known Terms: $\boxed{x}$

Terms to be estimated: $\boxed{\lambda, f, \epsilon, \mu}$

(d). Assumptions:

Latent variable (factor f) follows a gaussian dist.

$$f \sim \mathcal{N}(0, I)$$

while Mean $= 0$ and covariance $= I$.

In [62]:

```
# Pkg.add("Cairo");
using Distributions

mu = [10 20 30 40]';
Lambda =
[1.0 0 0
 0 1.0 0
 0 0 1.0
 0.5 0.5 0];
Psi = diagm([0.1, 0.2, 0.3, 0.4]);
d1 = MvNormal([0,0,0],ones(3));
X = zeros(50,4);
for i=1:50
f = rand(d1,1);
d2 = MvNormal(vec(mu+ Lambda*f),Psi);
x = rand(d2,1);
X[i,:] = x';
end
X
```

Out[62]:

```
50×4 Array{Float64,2}:
 10.9157   19.8912  31.1335  40.5303
 11.2593   20.2802  30.6414  42.1862
 11.6201   20.1413  28.482   39.8312
  9.38563  21.4925  29.369   40.9083
 10.1267   21.3495  29.3811  41.7121
  8.56655  21.3649  31.9531  39.7583
 10.8312   20.9982  30.7176  41.9568
 11.2732   20.8429  28.6785  40.8595
  7.76473  19.1212  29.1907  38.9001
 10.1661   20.8392  29.552   40.0082
 10.4591   21.4165  29.8694  39.7899
 10.5241   20.5505  27.0688  40.2748
 11.4772   20.6228  31.3478  39.9425
   ⋮
 10.938    20.1902  30.0361  40.0494
  9.87135  19.9888  29.6799  40.4248
  8.5961   20.3973  30.1045  40.4084
  8.05787  21.6147  30.914   40.4337
  9.81402  21.3347  30.7051  40.6769
  9.2221   21.2462  30.9303  40.511
  8.94407  18.3371  29.176   38.7042
 10.6674   19.4005  29.9369  41.2012
  9.58202  19.7752  28.6777  38.9778
 12.105    19.1915  29.1306  40.7783
  9.46215  20.3068  28.9354  40.3139
 11.6499   18.6145  30.4429  40.0936
```

In [63]:

```
function E_Step(X,mu,Lambda,Psi,k)
mu_f_by_x = (X - repmat(mu',size(X,1),1))*(Lambda'*inv(Lambda*Lambda' + Psi))';
Sig_f_by_x = eye(k) - Lambda'*inv(Lambda*Lambda' + Psi)*Lambda;
return mu_f_by_x,Sig_f_by_x;
end
```

Out[63]:

```
E_Step (generic function with 1 method)
```

In [64]:

```
function M_Step(X,mu_f_by_x,Sig_f_by_x,k)
nrows, ncols = size(X);
#Computing mu
mu = mean(X,1)';
#Computing Lambda
Lambda_term1 = zeros(ncols,k);
Lambda_term2 = zeros(k,k);
for i=1:nrows
Lambda_term1 = Lambda_term1 + ((X[i,:] - mu)*mu_f_by_x[i,:]');
Lambda_term2 = Lambda_term2 + (mu_f_by_x[i,:]*mu_f_by_x[i,:]')+Sig_f_by_x;
end
Lambda = Lambda_term1*Lambda_term2;
#Computing Psi
Phi = zeros(ncols,ncols);
for i=1:nrows
Phi = Phi + (X[i,:]*X[i,:]' - X[i,:]*mu_f_by_x[i,:]'*Lambda' - Lambda*mu_f_by_x[i,:
end
Psi = diagm(diag(Phi./nrows));
return mu, Lambda, Psi
end
function compute_llh(X,mu,Lambda,Psi)
llh = 0;
for i=1:size(X,1)
llh = llh + log(pdf(MvNormal(vec(mu),(Lambda*Lambda')+Psi),X[i,:]));
end
return llh;
end
```

Out[64]:

```
compute_llh (generic function with 1 method)
```

In [65]:

```
function fa_em(X,k)
    max_Iter = 100;
    eps = 0.0001;
    llh = -Inf*ones(max_Iter+1);
    mu = mean(X,1)';
    Lambda = rand(size(X,2),k);
    Psi = diagm(rand(size(X,2)));
    print(mu,"\n",Lambda,"\n",Psi,"\n");
    llh[1] = compute_llh(X,mu,Lambda,Psi);
    print(llh[1],"\n")
    for i=1:max_Iter
    print(i,"\n");
    mu_f_by_x,Sig_f_by_x = E_Step(X,mu,Lambda,Psi,k);
    mu_new, Lambda_new, Psi_new = M_Step(X,mu_f_by_x,Sig_f_by_x,k);
    print(mu_new,"\n",Lambda_new,"\n",Psi_new,"\n");
    llh[i+1] = compute_llh(X,mu_new,Lambda_new,Psi_new);
    print(llh[i+1],"\n");
    if(sum(abs.(mu_new-mu))<eps && sum(abs.(Lambda_new-Lambda))<eps && sum(abs.(Psi
        break;
    end
    mu = mu_new;
    Lambda = Lambda_new;
    Psi = Psi_new;
    end
    mu_f_by_x,Sig_f_by_x = E_Step(X,mu,Lambda,Psi,k);
    return (mu, Lambda, Psi, mu_f_by_x, Sig_f_by_x, llh);
end
```

Out[65]:

```
fa_em (generic function with 1 method)
```

In [66]:

```
#Calling the EM approach for dataset X and 3 factors
mu, Lambda, Psi, mu_f_by_x, Sig_f_by_x, llh = fa_em(X,3)
```

```
2
[10.2494; 20.1751; 29.7844; 40.2144]
[0.0707107 0.312041 0.101665; -0.0101316 -0.403127 0.649494; 0.7890
46 0.345488 0.0883719; 0.0648728 0.0200756 0.583952]
[106.267 0.0 0.0 0.0; 0.0 408.261 0.0 0.0; 0.0 0.0 889.044 0.0; 0.0
0.0 0.0 1617.91]
-805.6783651010933
3

[10.2494; 20.1751; 29.7844; 40.2144]
[1.86839 8.97183 2.18586; -0.307245 -3.77153 3.49144; 3.13611 1.036
4 0.508012; 0.810094 1.37568 2.26327]
[194.771 0.0 0.0 0.0; 0.0 434.415 0.0 0.0; 0.0 0.0 899.695 0.0; 0.0
0.0 0.0 1625.42]
-833.6340108912119
4
[10.2494; 20.1751; 29.7844; 40.2144]
[11.9823 63.1097 15.259; -1.66918 -21.7641 16.2585; 11.2748 -1.7190
9 1.54367; 4.42875 10.5447 11.4519]
[3053.95 0.0 0.0 0.0; 0.0 1021.21 0.0 0.0; 0.0 0.0 1019.12 0.0; 0.0
```

In [67]:

```
mu
```

Out[67]:

```
4×1 Array{Float64,2}:
 10.2494
 20.1751
 29.7844
 40.2144
```

The results match the parameter we used to generate data.

In [68]:

```
Lambda
```

Out[68]:

```
4×3 Array{Float64,2}:
  7.91435   76.0793   23.592
 -2.19493  -24.3011   20.9408
 28.615     -6.27952  -0.510796
  5.41899   12.6643   16.1697
```

In [69]:

```
Psi
```

Out[69]:

```
4×4 Array{Float64,2}:
 4364.21     0.0      0.0      0.0
    0.0   1064.51     0.0      0.0
    0.0      0.0   1424.77     0.0
    0.0      0.0      0.0   1899.73
```

In [70]:

```
llh
```

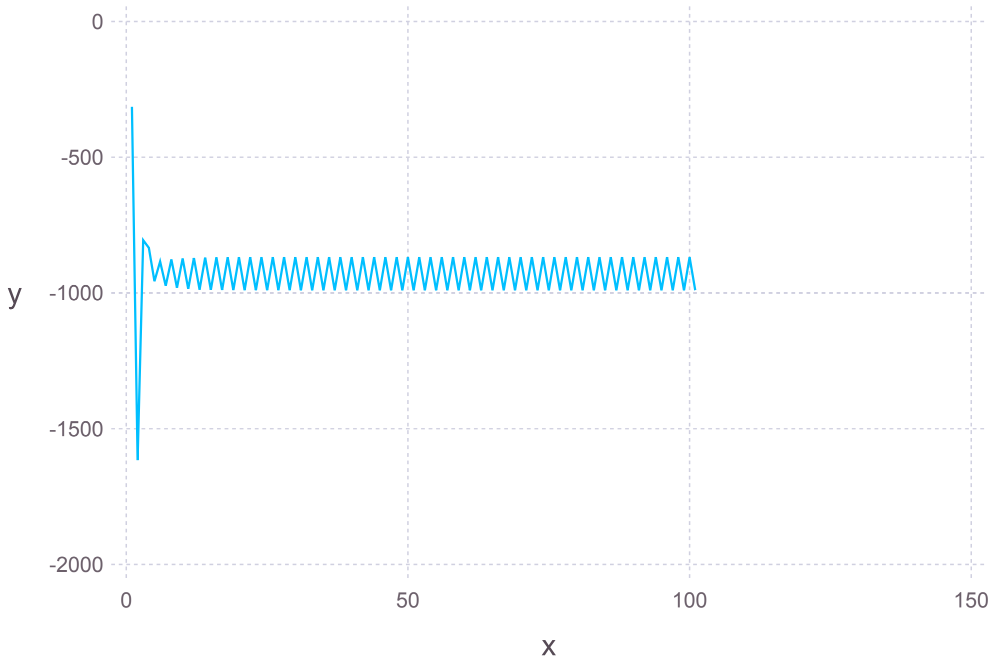
Out[70]:

```
101-element Array{Float64,1}:
  -314.078
 -1616.03
  -805.678
  -833.634
  -956.631
  -884.214
  -973.74
  -876.767
  -980.597
  -873.05
  -985.149
  -870.939
  -987.545
     ⋮
  -868.307
  -990.049
  -868.307
  -990.049
  -868.307
  -990.049
  -868.307
  -990.049
  -868.307
  -990.049
  -868.307
  -990.049
```

In [71]:

```
using Gadfly, Cairo, Fontconfig
#plot the log-likelihood
plot(x=collect(1:1:101), y=llh,Geom.line)
```

Out[71]:



The EM approach converges to the final esitimates. Our algorithm started converging just after 3-4 iteration.

For k = 2

In [55]:

```
mu, Lambda, Psi, mu_f_by_x, Sig_f_by_x, llh = fa_em(X,2)
```

```
0.0 0.0 1869.27]
-933.2207616082413
15
[9.99745; 20.0657; 30.0777; 40.0283]
[2.23754 7.3517; 26.9937 -9.11396; 29.6519 -19.701; 11.8225 -0.1246
41]
[121.118 0.0 0.0 0.0; 0.0 876.719 0.0 0.0; 0.0 0.0 1644.62 0.0; 0.0
0.0 0.0 1678.63]
-879.7501404402069
16
[9.99745; 20.0657; 30.0777; 40.0283]
[21.2759 47.8385; 15.47 0.979071; 10.0535 -19.7825; 13.2726 15.476]
[1903.31 0.0 0.0 0.0; 0.0 509.253 0.0 0.0; 0.0 0.0 1134.59 0.0; 0.0
0.0 0.0 1870.07]
-931.9414478262806
17

[9.99745; 20.0657; 30.0777; 40.0283]
[2.20779 7.35812; 27.8139 -9.46218; 30.6237 -20.1805; 12.1299 -0.24
9306]
```

In [56]:

```
mu
```

Out[56]:

```
4×1 Array{Float64,2}:
  9.99745
 20.0657
 30.0777
 40.0283
```

In [57]:

```
Lambda
```

Out[57]:

```
4×2 Array{Float64,2}:
 20.3356    48.3952
 13.9042     1.06843
  8.95359  -19.3295
 12.4187    15.6389
```

In [58]:

```
Psi
```

Out[58]:

```
4×4 Array{Float64,2}:
 1911.05    0.0      0.0      0.0
    0.0   486.47     0.0      0.0
    0.0     0.0   1124.35     0.0
    0.0     0.0      0.0   1858.45
```

In [59]:

```
llh
```

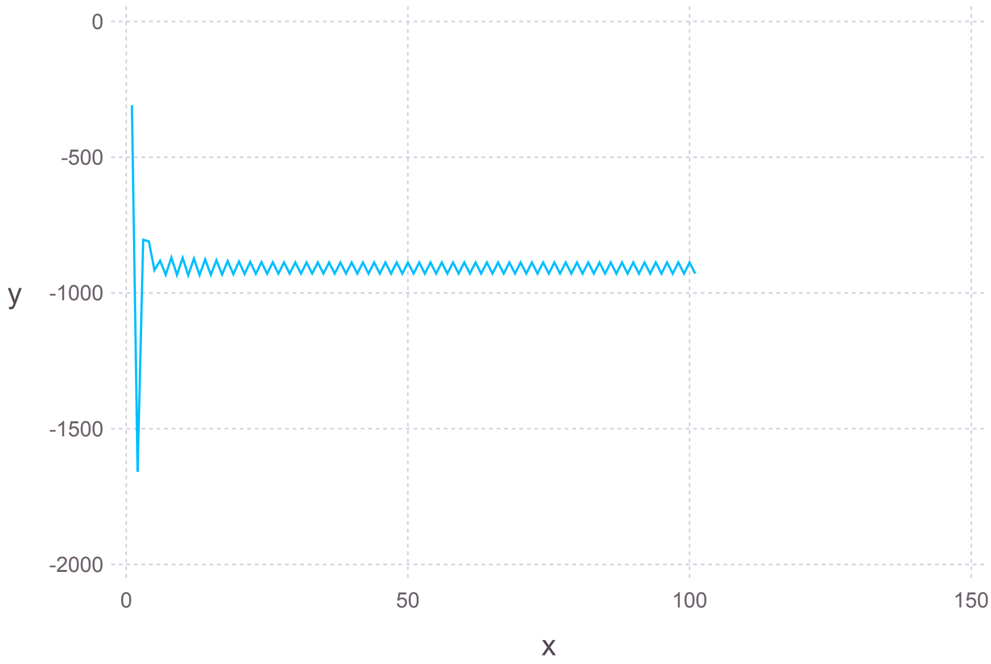
Out[59]:

```
101-element Array{Float64,1}:
  -308.159
 -1658.17
  -804.137
  -810.217
  -916.108
  -881.283
  -933.128
  -869.544
  -933.876
  -870.753
  -934.851
  -873.409
  -934.386
     ⋮
  -887.596
  -928.879
  -887.591
  -928.885
  -887.587
  -928.891
  -887.583
  -928.897
  -887.579
  -928.903
  -887.575
  -928.908
```

In [60]:

```
using Gadfly, Cairo, Fontconfig
#plot the log-likelihood
plot(x=collect(1:1:101), y=llh,Geom.line)
```

Out[60]:



The mean value mu is almost the same for both k = 2 and 3 and they match the mean parameter we used to generate the data. For k=3, Lambda has dimension 4 *3 and for k=2, Lambda has dimension 4* 2 as we only consider 2 factors for the second part. The values of Lambda and psi have different values in both analysis because there could be many possible solutions (indeterminancy problem).

In [ ]:

```

```