# Example using BRCA

## 1. Step 1 : Data loading and processing

Note that this step is also a part of "main_real.m" file.

The following matlab file can be downloaded from https://www.dropbox.com/s/v22fx0j2gnpeta6/all_data.mat?dl=0

```
load('all_data')
```

- all_clin includes clinical information for the patients from 22 cancer types
- all_exp is the rna data for all the patients
- all_mirna is the mirna data for all the patients
- all_cna is the cna data for all the patients
- all_pat is the index vector of patients indicating corresponding cancer types (See the first column of all_clin for the original cancer name)

### 1-1. Step 1-1: Consider a breast cancer (BRCA)

```
iii=1;
```

Here, “iii” is the cancer index from the 22 cancer types of the index (e.g. iii=1 represents BRCA). The followings are indices of each of the 22 cancer types:

- iii = 1 BRCA
- iii = 2 STAD
- iii = 3 LUAD
- iii = 4 LUSC
- iii = 5 COAD
- iii = 6 HNSC
- iii = 7 KIRC
- iii = 8 BLCA
- iii = 9 UVM
- iii = 10 PRAD
- iii = 11 SARC
- iii = 12 KIRP
- iii = 13 LIHC
- iii = 14 PAAD
- iii = 15 ESCA

- iii = 16 LGG
- iii = 17 MESO
- iii = 18 UCEC
- iii = 19 THCA
- iii = 20 READ
- iii = 21 OV
- iii = 22 CESC

### 1-2. Step 1-2: Choose patients belonging to the group of cancer with index iii and conduct data processing

```
iii_set= setdiff(1:33, [10 11 12 16 20 24 25 27 29 31 33]);

iii=iii_set(iii);
ind_set=find(all_pat==iii);
n=length(ind_set);
sel_exp=(all_exp(:,ind_set));
sel_cna=(all_cna(:,ind_set));
sel_mirna=(all_mirna(:,ind_set));
sel_clin=all_clin(ind_set,:);
```

Here, “iii_set” is the set of indices of cancer types in the raw data set.
Now, we extract survival time and statuses:

```
surv_stat=sel_clin(:,end-4);
surv_stat=strrep(table2array(surv_stat),'Alive','1');
surv_stat=strrep(surv_stat,'Dead','0');
surv_stat=str2double(surv_stat);
surv_time=sel_clin(:,end-3);  surv_time=table2array(surv_time);
surv_time=str2double(surv_time);
```

We choose patients having information about survival time and status:

```
ind_set2=setdiff(1:n,union(find(isnan(surv_time)), find(isnan(surv_stat))));
n=length(ind_set2);
surv_time=surv_time(ind_set2);  surv_stat=surv_stat(ind_set2);
```

We write omics data sets and clinical information and perform PCA (this step is optional):

```
sel_exp=(sel_exp(:,ind_set2));
sel_cna=(sel_cna(:,ind_set2));
sel_mirna=(sel_mirna(:,ind_set2));
sel_clin=sel_clin(ind_set2,:);

sel_exp2 = reg_pca(sel_exp',min(n,100));
sel_mirna2 = reg_pca(sel_mirna',min(n,100));
sel_cna2 = reg_pca(sel_cna',min(n,100));
```

## 2. Step 2: Conduct a proposed clustering algorithm

This step is also a part of "main_real.m" file.

### 2-1: Step 2-1 : Generate similarity matrices for clustering analysis

```
K=3;
gg=ones(1,K);
sigma_set=1:0.25:2;
g_set=10:2:30;
data_set3={sel_exp2,sel_mirna2,sel_cna2};
```

Here, 'K' is a number of omics data, 'sigma_set' and 'g_set' are sets of 'sigma' and 'g' that are parameters of Gaussian kernels, respectively.

### 2-2: Step 2-2 : Construct multiple similarity matrices using the function "generate_sim_matrices"

```
[Wfc0s_euc_near_n]=generate_sim_matrices(K,data_set3,gg,0,sigma_set,g_set);
```

Here, the function “generate_sim_matrices.m” can be found in the “Main_functions” directory.

## 2-3: Step 2-3 : Run the proposed clustering algorithm

```
CCC=4; c=0.1; rho=2; lam=0.001; mu=1; eta=1;
```

Here, “CCC” is a target clustering number and “c”, “rho”, “lam”, “mu”, “eta” are regularization parameters.

Now, we solve the optimization problem and get learned weight for each omic data:

```
[P_set,V_set,V_tot,ck,W_set,Wg_set] = clus_sim_update(CCC,c,rho, n, K, 5,11,gg, lam, mu, eta, Wfc0s_euc_near_n);

ck123=ck; ck_set={ck123};
tresult=[P_set,V_set,V_tot,ck,W_set,Wg_set];  tresult_final=tresult;
```

Here, the function "clus_sim_update.m" can be found in the "Main_functions" directory, and “ck123" records the weight of each data

Next, we incorporate learned weights for clustering analysis:

```
V_tot=[];
for dd=1:K;
   V_tot=[V_tot, tresult{K+1}(dd)*tresult{K+2}(:,(dd*CCC-CCC+1):(dd*CCC))];
end
V_tot=V_tot./ repmat(sqrt(sum(V_tot.^2,2)),1,size(V_tot,2));
```

Finally, we run K-means on the final output:

```
Clus_ind_wd123=litekmeans(V_tot,CCC,'Replicates',50);
```

Here, “Clus_ind_wd123” is the inferred cluster label.

## 3. Step 3 : Conduct a survival analysis

First, we define some variables that will be used to draw a graph:

```
name_i=1;
gp_title11={'KM Plot and fitted curve using the Weibull distribution'};
gp_title22={'Clus1','Clus2','Clus3','Clus4','Clus5', 'Clus6', 'ALL'};
colors_set={'-b','-g','-r','-k','-m', '--b','--g','--r','--k','--m'};
cl_title={'Blue', 'Green', 'Red', 'Black', 'Magenta','Blue--', 'Green--', 'Red--', 'Black--', 'Magenta--' };
```

We get patients' index set corresponding to each cluster:

```
ind_set=cell(1,CCC);
for ii=1:CCC
   ind_set{ii}=find(Clus_ind_wd123==ii);
end
```

We get sets of survival times and stauts for each cluster:

```
bbm_set=cell(1,CCC); bb_surv=cell(1,CCC);
   for ijk=1:CCC
      bbm_set{ijk}= surv_time(ind_set{ijk});
      bb_surv{ijk}= surv_stat(ind_set{ijk}) ;
   end
```

To use the function "generate_surv_func_general.m", we transform these data and generate a monthly survival time for each cluster:

```
bbm_set_set{1}=bbm_set; bb_surv_set{1}=bb_surv;

for kkkk=1:CCC
  bbm_set_set_m{1}{kkkk}=bbm_set_set{1}{kkkk}/30;
end;

for kkkk=1:CCC
  bbm_set_set_m{1}{kkkk}=bbm_set_set_m{1}{kkkk}.*(bbm_set_set_m{1}{kkkk}>0);
end
```

Then, we generate a monthly survival time for all patients in the chosen cancer type:

```
bbm_set_set_mt{1} = [];    bb_surv_sett{1} = [];
for kkkk=1:CCC
   bbm_set_set_mt{1}=[bbm_set_set_mt{1}, bbm_set_set_m{1}{kkkk}'];
   bb_surv_sett{1} = [bb_surv_sett{1}, bb_surv_set{1}{kkkk}'];
end
```

### 3-1. Step 3-1 : Fit Weibull survival model to each inferred cluster

First, set a ”bord” value, which is an upper limit of survival time (month) when generating a graph. Then, we generate survival curves for each cluster:

```
bord=120;
[pd1_A_set,pd1_B_set]=generate_surv_func_general(CCC, bord, bbm_set_set_m,bb_surv_set, 1, colors_set,gp_title11,gp_title22,cl_title, name_i)
```

### 3-2. Step 3-2: Compute the metrics that measure heterogeneity of survival times between inferred groups

Compute the metrics:

```
diff_area_set=diff_area_func(CCC, pd1_A_set, pd1_B_set, bord, 0.01);

diff_area_sum =sum(sum(diff_area_set));
diff_area_min =min(min(diff_area_set+10000*diag(ones(1,CCC))));

diff_avg_set=diff_area_func_ave(CCC, pd1_A_set, pd1_B_set, bord, 0.01);
diff_avg_sum=sum(sum(diff_avg_set))/CCC; diff_avg_min=min(diff_avg_set);
diff_area_set2=diff_area_func2(CCC, pd1_A_set, pd1_B_set, bord, 0.01);
```

Now we are ready to compute “Area_Min” measure:

```
area_p=diff_area_set2./diff_area_set; area_p(isnan(area_p))=0;
area_prop_min =min(min(area_p+diag(ones(1,CCC))));
```

Here, "area_prop_min" corresponds to the "Area_Min" value. We can see that the obtained value is 1 as in the below:

```
area_prop_min =

    1
```

### 3-3. Step 3-3: Perform log-rank test

For log-rank test, we use the funtion “MatSurv”. Note that "MatSurv" is like a matlab version of "survminer R-package”.

Note that in the corresponding paper, we use the R for log-rank test, but this matlab version also gives similar results. First, generate a group index "GroupVar" as follows (this should be cell-type):

```
gp_index = [{'1'}, {'2'}, {'3'}, {'4'}, {'5'}, {'6'}, {'7'}, {'8'}]

GroupVar = [];
for kkkk=1:CCC
   GroupVar = [GroupVar, repmat({gp_index{kkkk}}, 1, length(bbm_set_set_m{1}{kkkk}))];
end
```

"p" is a log-rank p-value

```
[p,fh,stats]=MatSurv(bbm_set_set_mt{1}', bb_surv_sett{1}', GroupVar', 'GroupsToUse', {'1','2','3','4'});
p
```

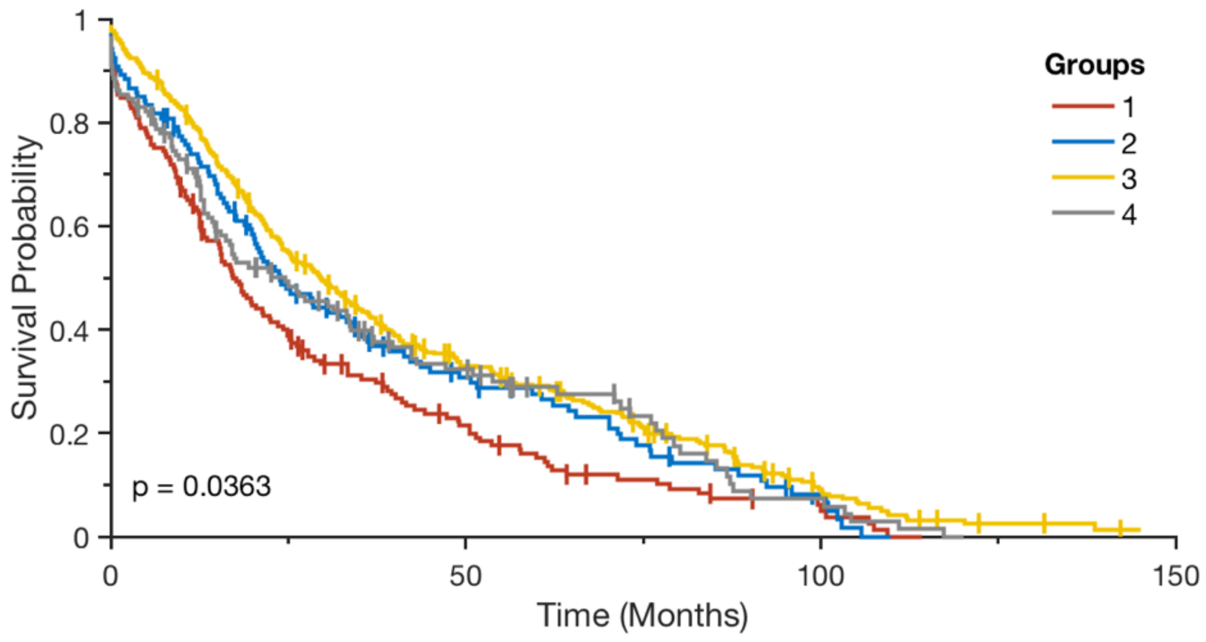
We get the log-rank p-value 0.0363:

```
 p

p =

   0.0363
```

We also obtain survival curves for each group:


Groups
Survival Probability
Time (Months)
p = 0.0363