ICeDT

Immune Cell Deconvolution in Tumor tissues

Update for version 0.99.1, 2019 02/25 to 03/07

Nomenclature

- C: variables for Consistent genes, CM or M was used in some cases.
- A: variables for Abberant genes
- Use lower case at the begining of the variable name. For example Sigma2C is changed to sigma2C.
- Y: observed expression data from bulk (mixed) samples
- X: observed expression data from purified samples, assume at least two samples per cell type.
- Z: estimated cell type-specific expression for each cell type. Previously there were notations such Z_0 and Z_1 and they are the same. Now the notations are unified to be Z. Prevoiusly Z includes expression from tumor, which are unknown in practice or assumed to be 0. In this update, Z does not includes expression from tumor.

Simulation codes

1. Move the code Simulation_Machinery.R into the R package as function simFunc.
2. Rename "Simulation_Machinery.RData" to "mean_var_relation.RData" and move it into the R package.
3. Remove functions weight_creation, meanFun, and IQRFun.
4. Modity the function to set muX of tumor to be -20. Prevoiusly muX was set to be the same value as other cell types, and then the simulated gene expression from tumor was set to 0. This has two consequence

- the ouptut of `tumor_mu` is wrong since it is muX
- for any abberant genes, the gene expression from tumor is not 0.

ICeDT algorithm

1. Combine the codes in *_initFit.R, *_UpdateFunctions.R, and *_FittingAlgorithm.R into one file.
2. Combine HS2_UpdateWgts_All_* and HS2_UpdateWgts_Single_*, to write HS2_UpdateWgts_Single_* within a for loop. Prevoiusly apply was used. For a complex function like that, apply will not be faster than for loop.
3. Change function names or parameter names

- change function name ICeDT_fit_noWgt_noRef to ICeDT_noWgt_noRef, remove parameter Subj_CO, and change parameter RhoConv_CO to rhoConverge.
- HS2_UpdateWgts_* to updateWgts_
- HS_* to HS
- p_m or Pm or to propC, which is the mixture proportio of consistent genes.

4. Z_star was only used in function HS_GradFunc_Fix, but it is one of the parameters for several other functions of gradient and liklihood. Remove it as a parameter and calculate within the function of HS_GradFunc_Fix

Strcuture of function ICeDT_noWgt_noRef

1. Check input
2. Given observed cell type-specific gene expression of multiple samples per cell type, calculate cell type-specific expression per cell type.
3. Call `PropPlus_Update` to update rho, sigma2C, sigma2A, and propC.

- Iteratively update weights (each gene's probability being consistent) using function `updateWgts` (E-step) and estimate three parameters: cell type proportions, sigma2c, and sigma2A using function `updatePropn_All` (M-step).
  - `updatePropn_Single` iteratively estimate sigma2c, sigma2A, and cell type compositions. The latter were estimated using Augmented Lagrangian Minimization Algorithm implmented in function `alabama/auglag`.

4. Update weights one more time

Handeling the special cell type (fixed cell type) of tumor cells.

Prevoius approach

1. When estimating cell type-specific gene expression, force gene expression from tumor cells to be 0. No matter what is the observed gene expression from tumor cells.

```
  Z     = exp(CT_MU + CT_var/2)
  Z[,1] = rep(0, nG)
```

2. `PropPlus_Update` takes initial estimates of rho as relative proportions from all the other cell types other than tumor cells.
3. In function `updateWgts`, add 0 as the proportion from tumor to the rho vector.

```
updateWgts <- function(logY, rho_init, sigma2C, sigma2A, Z, propC){

  EM_wgt = matrix(NA, nrow=nrow(logY), ncol=ncol(logY))

  for(i in 1:ncol(logY)){

    logY_i      = logY[,i]
    rho_init_i = rho_init[,i]

    #-----------------------------------------------#
    # Cmu_ij for Consistent Marker Gene Probs       #
    # Amu_ij for Aberrant Marker Gene Probs         #
    #-----------------------------------------------#

    eta_ij = Z %*% matrix(c(0, rho_init_i), ncol=1)
    Cmu_ij = log(eta_ij) - sigma2C[i]/2
    Amu_ij = log(eta_ij) - sigma2A[i]/2
    C_lLik = dnorm(logY_i, mean = Cmu_ij, sd = sqrt(sigma2C[i]), log = TRUE)
    A_lLik = dnorm(logY_i, mean = Amu_ij, sd = sqrt(sigma2A[i]), log = TRUE)

    EM_wgt[,i] = 1/(1+((1-propC[i])/propC[i])*exp(A_lLik-C_lLik))
  }
```

```r
  return(EM_wgt)
}
```

4. In function `updatePropn_Single`, always calculate expected expression in bulk tumor by assuming proportion from tumor is 0.

```r
  eta_ij     = drop(Z %*% matrix(c(0,urho_1), ncol=1))
  log_eta_ij = log(eta_ij)

  sigma2C_1  = sigma2_Update(logY = logY, log_eta_ij = log_eta_ij,
                             EM_wgt = EM_wgt, AB_Up = FALSE)
  sigma2A_1  = sigma2_Update(logY = logY, log_eta_ij = log_eta_ij,
                            EM_wgt = EM_wgt, AB_Up = TRUE)

  mu_ijC = log_eta_ij - sigma2C_1/2
  mu_ijA = log_eta_ij - sigma2A_1/2

  logLik_1 = sum(EM_wgt*dnorm(logY,mean=mu_ijC,sd=sqrt(sigma2C_1),log=TRUE))+
    sum((1-EM_wgt)*dnorm(logY,mean=mu_ijA,sd=sqrt(sigma2A_1),log=TRUE))
```

When estimating rho and `useRho` is TRUE, first estimate rho given proportion of tumor cells, and the normalize it so that its summation is 1.

```r
    if(useRho){
      auglagOut = auglag(par = urho_0[-c(nCell)], fn = logLik_Fix,
                         gr = gradFunc_Fix, hin = hin_func_Fix,
                         hin.jac = hin_jacob_Fix, logY = logY,
                         rho_i0 = rho_i0, Z=Z, sigma2C = sigma2C_0,
                         sigma2A = sigma2A_0, EM_wgt = EM_wgt,
                         control.optim = list(fnscale=-1),
                         control.outer = list(trace=FALSE))

      urho_1[c(1:(nCell-1))] = auglagOut$par
      urho_1[nCell] = 1-rho_i0-sum(auglagOut$par)

      urho_1 = correctRho(est = urho_1, total = 1-rho_i0)
    }
```

The objective function for optimation do consider the gene expression from tumor cells. This is the only place where gene expressoin from tumor cells (1st column of Z) is used, but it was assumed cell type proprtion of tumor is 0. In addition, in function `ICeDT_noWgt_noRef`, `Z[,1]` is initilized by 0,

```r
logLik_Fix <- function(x, logY, rho_i0, Z, sigma2C, sigma2A, EM_wgt){
  rho   = c(0, x, 1-rho_i0-sum(x))

  eta_ij = Z %*% rho
  mu_ijC = log(eta_ij) - sigma2C/2
  mu_ijA = log(eta_ij) - sigma2A/2

  out = sum(EM_wgt*dnorm(logY, mean = mu_ijC, sd = sqrt(sigma2C), log = TRUE)) +
    sum((1-EM_wgt)*dnorm(logY, mean = mu_ijA, sd = sqrt(sigma2A), log = TRUE))

  return(out)
}
```

New approach

1. Change the dimeion of Z so that it does not inlcude gene expression from tumor cells. We have to assume for those marker genes, the expression from tumor cells are 0 anyway.
2. Modify those codes mentioned above to remove tumor purity from `rho`, so that `eta_ij = Z %*% rho` has correct dimension.
3. Modify the code so that the observed cell type-specific expression data X does not include gene expression data from tumor cells.
4. Remove the paramerer `fixed_CT` that specifies which cell type is tumor cell.

Compare functions in noWgt_noRef vs. noWgt_suppRef_*

1. `lmInit` = `lmInit_noWgt_suppRef`
2. `sigmaInit` = `AbProf_Init_noWgt_suppRef`, except calling function `sigma2_Update` or `HS_Sigma2_Update_noWgt_supp`
3. `sigma2_Update` = `HS_Sigma2_Update_noWgt_suppRef`
4. `gradFunc_noFix` = `HS_GradFunc_noFix_noWgt_suppRef`
5. `gradFunc_Fix` = `HS_GradFunc_Fix_noWgt_suppRef`
6. `logLik_noFix` = `HS_ComputeLik_noFix_noWgt_suppRef`
7. `logLik_Fix` = `HS_ComputeLik_Fix_noWgt_suppRef`
8. `updatePropn_Single` = `HS_UpdatePropn_Single_noWgt_suppRef`
9. `updatePropn_All` = `HS_UpdatePropn_All_noWgt_suppRef`
10. `updateWgts` = `HS2_UpdateWgts_All_noWgt_suppRef`
11. `PropPlus_Update` = `PropPlus_Update_noWgt_suppRef`

The only difference between noWgt_noRef and noWgt_suppRef is that the former first estimate sigmature matrix from observed cell type-specific gene expression data. So **remove all the redundant functions and just add an parameter to indicate whether the input is cell type-specific gene expression or signature matrix**.

Compare functions in noWgt_noRef vs. Wgt_suppRef_*.

The function names have been updated in the main function, e.g., `gradFunc_noFix` becomes `gradFunc_noPurity`.

1. `lmInit` = `lmInit_Wgt_suppRef`
2. `sigmaInit` = `AbProf_Init_Wgt_suppRef`, except calling function `sigma2_Update` or `HS_Sigma2_Update_Wgt_suppRef`.
3. `sigma2_Update` is not the same as `HS_Sigma2_Update_Wgt_suppRef` since the latter using numerical optimzation. Modify `sigma2_Update` to accomodate this.
4. `HS_ComputeLik_Fix_Wgt_suppRef`, `HS_GradFunc_Fix_Wgt_suppRef`, `HS_ComputeLik_noFix_Wgt_suppRef`, and `HS_ComputeLik_Fix_Wgt_suppRef` take both sigma2 and var_wgt as their parametrs and multiply them inside the function to generate the actual variance, e.g., `Sigma2_wgt  = var_wgt*Sigma2`. In the new functions, we simply provide the variance after multiplying weigths, so that they are the same as the cases without weights.
5. `updatePropn_Single` = `HS_UpdatePropn_Single_Wgt_suppRef`, after we update variance e.g..

```
  sigma2C_1s = sigma2_Update(logY = logY, log_eta_ij = log_eta_ij,
                             EM_wgt = EM_wgt, AB_Up = FALSE, var_wgt=var_wgt)
  sigma2A_1s = sigma2_Update(logY = logY, log_eta_ij = log_eta_ij,
                             EM_wgt = EM_wgt, AB_Up = TRUE, var_wgt=var_wgt)

  if(! is.null(var_wgt)){
    sigma2C_1 = var_wgt*sigma2C_1s
    sigma2A_1 = var_wgt*sigma2A_1s
  }else{
    sigma2C_1 = sigma2C_1s
    sigma2A_1 = sigma2A_1s
  }
```

9. `updatePropn_All` = `HS_UpdatePropn_All_noWgt_suppRef`, after adding `var_wgt` as a parameter.
10. `updateWgts` = `HS2_UpdateWgts_All_noWgt_suppRef`, after adding `var_wgt` as a parameter, and update sigma2 if `!is.null(var_wgt)`.
11. `PropPlus_Update` = `PropPlus_Update_noWgt_suppRef`, after adding `var_wgt` as a parameter.

To be fixed.

1. function `alabama/auglag` gives warning message when choosing parameters outside the constraints, for example, setting rho to be larger than 1.
2. now the convergence criterion is that all samples converge. This is not necessary since the computation is independent across samples.