TimeMeter: An R package for assessing temporal gene expression similarity and identifying differentially progressing genes

Peng Jiang
Regenerative Biology Laboratory,
Morgridge Institute for Research
Madison, WI 53707, USA

03/21/2020

Contents

Introduction

TimeMeter is a statistical tool and R package to assess temporal gene expression pattern similarity, and identify differential progressing genes. TimeMeter uses the dynamic time warping (DTW) algorithm to align two temporal sequences. TimeMeter first post-processes DTW alignment by truncating certain start or end points based on alignment patterns. This will result in a truncated alignment which represents the time frames from each time series that are comparable. Then it calculates four measurements that jointly assess gene pair temporal similarity: percentage of alignment for (1) query and for (2) reference, respectively; (3) aligned gene expression correlation; (4) likelihood of alignment arising by chance. These four novel metrics in TimeMeter give temporal similarity assessments on different aspects, and the joint requirement of these four metrics will give a high confident assessment for the temporal pattern similarity. For gene pairs with similar temporal patterns, TimeMeter partitions the temporal associations into separate segments via piecewise regression.

The differential progression between gene pairs is calculated by aggregation of progression difference in each segment.

Methods

TimeMeter Algorithm: assessing temporal similarity

TimeMeter first uses dynamic time warping (DTW) algorithm to align two time-course gene expression vectors (a query and a reference; length may vary) via the R package ("dtw"). TimeMeter corrects the DTW aligned indices by truncating the first (m-1) start time points in one gene if the first m time points can be aligned to the same start points in another gene, and terminating alignment (truncating the rest of time points) if DTW matches the last elements in any of the genes. This will exclude certain time points from alignment, and result in a truncated alignment. TimeMeter then calculates four metrics on truncated alignment that jointly assess gene pair temporal similarity: percentage of alignment for (1) query and for (2) reference, respectively; (3) aligned gene expression correlation; (4) likelihood of alignment arising by chance:

1. Percentage of alignment for the query: the length of aligned time interval (after truncation) in query divided by total length of query time interval.
2. Percentage of alignment for the reference: the length of aligned time interval (after truncation) in reference divided by total length of reference time interval.
3. Aligned gene expression correlation (Rho): it is calculated by the Spearman's rank correlation coefficient (Rho) for aligned gene expressions (after truncation). It measures how well the gene expression correlations are after alignment.
4. Likelihood of alignment arising by chance (p-value): To further rule out the alignment is not raised by chance, for each gene pair, TimeMeter shuffles the gene expression values of both query and reference separately for 100 times. For each shuffling, it calculates the aligned gene expression Spearman correlation coefficient (Rho) (measurement 2). TimeMeter assumes that the Rho from shuffling follows a Gaussian distribution with mean (μ) and standard deviation (σ). It calculates the p-value of likelihood of alignment arising by chance by lower-tail probability of Gaussian distribution (μ, σ^2), assuming the aligned gene expression correlations between query and reference should be significantly higher than these shuffled temporal gene expressions.

These four measurements will give temporal similarity assessments on different aspects.

TimeMeter Algorithm: identifies differential progression genes

Given a STP gene pair (identified by previous step), TimeMeter scores the progression difference based on truncated alignment. For each query time point within truncated alignment, TimeMeter groups and calculates the average corresponding aligned reference time. This will result in two variables: aligned query time as independent variable and average aligned reference time as dependent variable. Next, TimeMeter applies piecewise (segmented) regression to these two variables, and partitions them into separate segments. The breakpoints in piecewise regression are determined by the lowest Bayesian Information Criterion (BIC) via enumerating all K (K<=N) possible number of breakpoints (N =10 by default). For each segment, the slope of the regression measures the fold-change of the speed (query versus reference). A slope being greater or less than 1, indicates faster or slower dynamical speed of change, respectively. A slope equivalent or close to 1 is a special case in which the dynamical speed is the same or similar (time shift pattern). TimeMeter further merged adjacent segments if the absolute slope difference less than deltaSlope (we set deltaSlope = 0.1 for this study) by a linear regression, and recalculates the slope. This process is repeated until no adjacent segments have absolute slope difference less than deltaSlope. Then for each segment, TimeMeter calculates the area difference between under the segmented regression line and under the diagonal line, assuming that if the query and reference have no progression difference along time points, the aligned time points should follow the diagonal line (the aligned query time equals the aligned reference time). The extent of deviation from the diagonal line can be used to measure the progression difference. A progression advance score (PAS) is calculated by aggregation of area difference in each segment and normalized by total aligned time length (after truncation) in query.

Quick Start

Step 1: Loading the data (normalized gene expression values)

```
library("TimeMeter")
data(simData)
data=simdata$TimeShift_10
gene=data$gene
query=data$query
query
```

```
##   [1]  0.000000000  0.000000000  0.000000000  0.000000000  0.000000000
##   [6]  0.000000000  0.000000000  0.000000000  0.000000000  0.000000000
##  [11]  0.063391810  0.126528621  0.189156463  0.251023411  0.311880600
##  [16]  0.371483228  0.429591541  0.485971792  0.540397190  0.592648804
##  [21]  0.642516449  0.689799528  0.734307842  0.775862355  0.814295909
##  [26]  0.849453903  0.881194913  0.909391257  0.933929513  0.954710977
##  [31]  0.971652051  0.984684590  0.993756170  0.998830299  0.999886566
##  [36]  0.996920723  0.989944700  0.978986558  0.964090378  0.945316079
##  [41]  0.922739183  0.896450508  0.866555800  0.833175314  0.796443324
##  [46]  0.756507588  0.713528750  0.667679694  0.619144853  0.568119460
##  [51]  0.514808768  0.459427224  0.402197603  0.343350116  0.283121479
##  [56]  0.221753968  0.159494436  0.096593328  0.033303666 -0.030119962
##  [61] -0.093422430 -0.156349101 -0.218646847 -0.280065071 -0.340356714
##  [66] -0.399279251 -0.456595659 -0.512075381 -0.565495246 -0.616640368
##  [71] -0.665305014 -0.711293425 -0.754420611 -0.794513090 -0.831409587
##  [76] -0.864961683 -0.895034413 -0.921506807 -0.944272379 -0.963239552
##  [81] -0.978332030 -0.989489101 -0.996665887 -0.999833518 -0.998979251
##  [86] -0.994106524 -0.985234937 -0.972400176 -0.955653871 -0.935063385
##  [91] -0.910711543 -0.882696304 -0.851130359 -0.816140686 -0.777868033
##  [96] -0.736466353 -0.692102188 -0.644953996 -0.595211434 -0.543074594
## [101] -0.488753200 -0.432465763 -0.374438704 -0.314905441 -0.254105449
## [106] -0.192283301 -0.129687681 -0.066570383 -0.003185302
```

```
timePoints_query=data$timePoints_query
timePoints_query
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
##  [18]  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34
##  [35]  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51
##  [52]  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68
##  [69]  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85
##  [86]  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102
## [103] 103 104 105 106 107 108 109
```

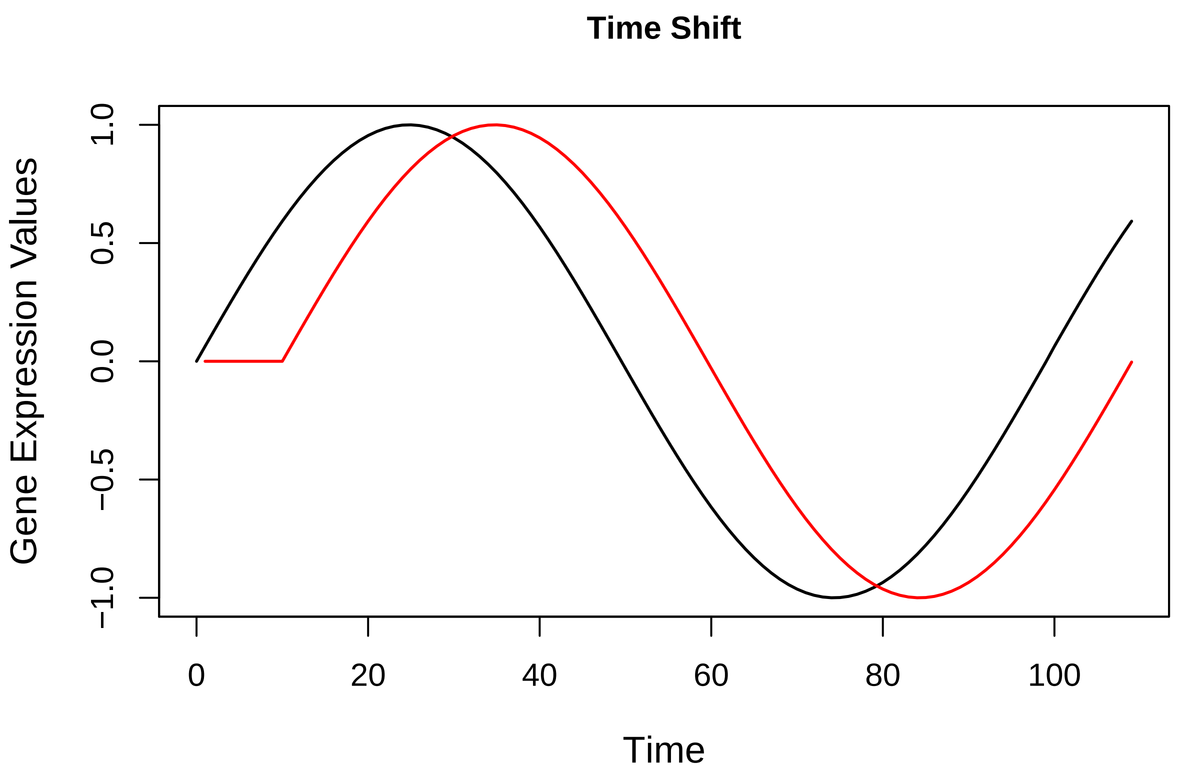
```
reference=data$reference
reference
```

```
##   [1]  0.000000000  0.063391810  0.126528621  0.189156463  0.251023411
##   [6]  0.311880600  0.371483228  0.429591541  0.485971792  0.540397190
##  [11]  0.592648804  0.642516449  0.689799528  0.734307842  0.775862355
##  [16]  0.814295909  0.849453903  0.881194913  0.909391257  0.933929513
##  [21]  0.954710977  0.971652051  0.984684590  0.993756170  0.998830299
##  [26]  0.999886566  0.996920723  0.989944700  0.978986558  0.964090378
##  [31]  0.945316079  0.922739183  0.896450508  0.866555800  0.833175314
##  [36]  0.796443324  0.756507588  0.713528750  0.667679694  0.619144853
##  [41]  0.568119460  0.514808768  0.459427224  0.402197603  0.343350116
##  [46]  0.283121479  0.221753968  0.159494436  0.096593328  0.033303666
##  [51] -0.030119962 -0.093422430 -0.156349101 -0.218646847 -0.280065071
##  [56] -0.340356714 -0.399279251 -0.456595659 -0.512075381 -0.565495246
##  [61] -0.616640368 -0.665305014 -0.711293425 -0.754420611 -0.794513090
##  [66] -0.831409587 -0.864961683 -0.895034413 -0.921506807 -0.944272379
##  [71] -0.963239552 -0.978332030 -0.989489101 -0.996665887 -0.999833518
##  [76] -0.998979251 -0.994106524 -0.985234937 -0.972400176 -0.955653871
##  [81] -0.935063385 -0.910711543 -0.882696304 -0.851130359 -0.816140686
##  [86] -0.777868033 -0.736466353 -0.692102188 -0.644953996 -0.595211434
##  [91] -0.543074594 -0.488753200 -0.432465763 -0.374438704 -0.314905441
##  [96] -0.254105449 -0.192283301 -0.129687681 -0.066570383 -0.003185302
## [101]  0.063391810  0.126528621  0.189156463  0.251023411  0.311880600
## [106]  0.371483228  0.429591541  0.485971792  0.540397190  0.592648804
```

```
timePoints_reference=data$timePoints_reference
timePoints_reference
```

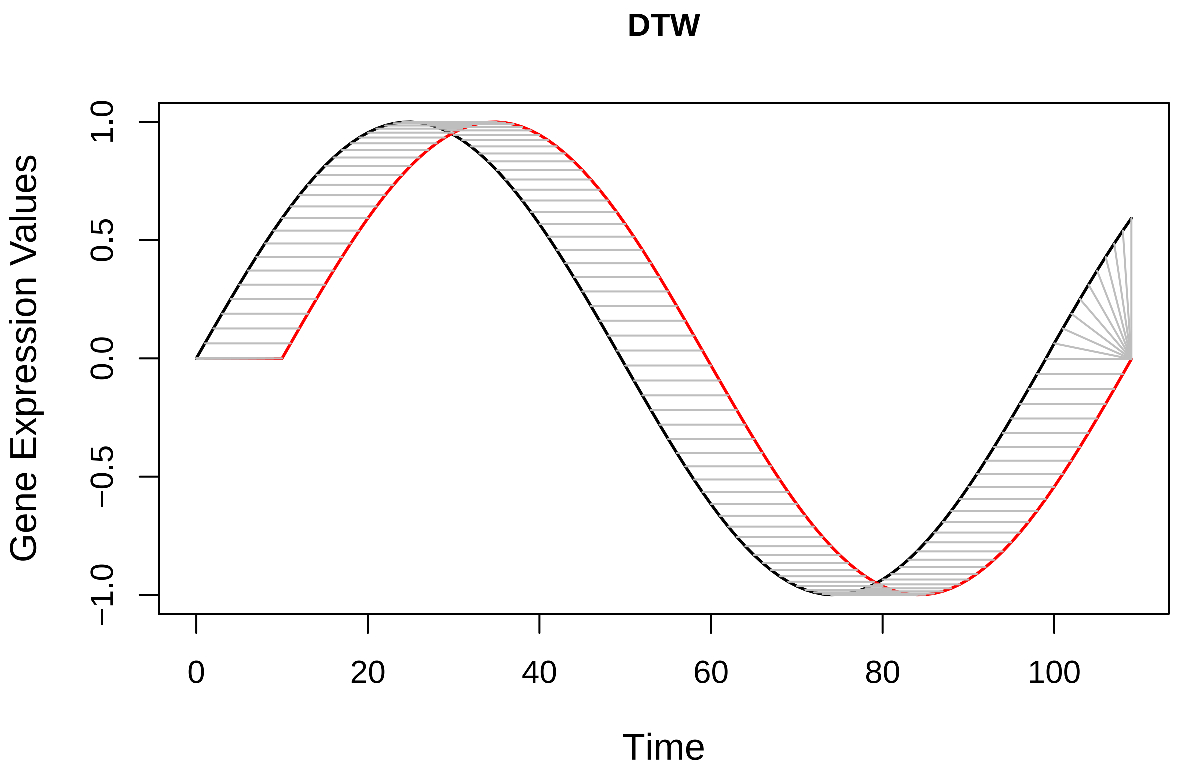
```
##   [1]   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16
##  [18]  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33
##  [35]  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50
##  [52]  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67
##  [69]  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84
##  [86]  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101
## [103] 102 103 104 105 106 107 108 109
```

```
plotExpOriginal(query,timePoints_query,reference,timePoints_reference,
                title="Time Shift",ref_col="black",query_col="red")
```



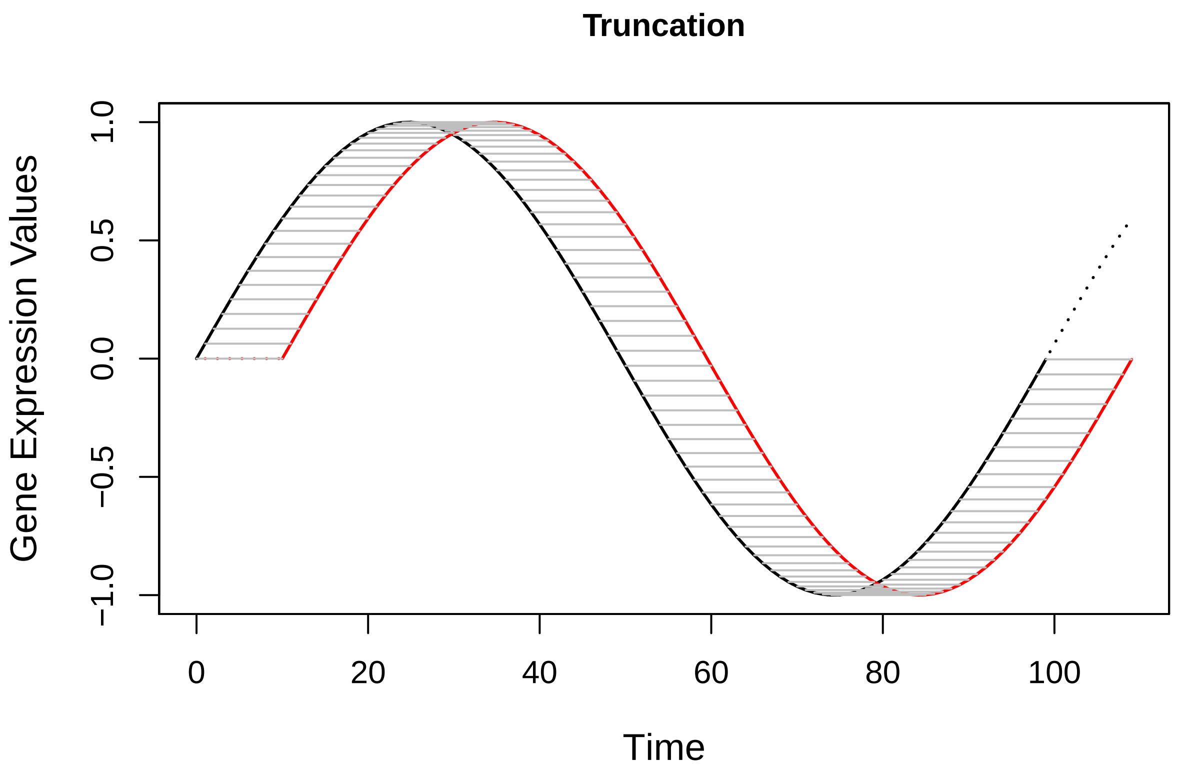
Step 2: Temporal Alignment via DTW

```r
alignment=dtw(query,reference)
dtw_results=list(alignment$index1,alignment$index2)
index_1=dtw_results[[1]]
index_2=dtw_results[[2]]
aligned_values_query=query[index_1]
aligned_values_reference=reference[index_2]
aligned_timePoints_query=timePoints_query[index_1]
aligned_timePoints_reference=timePoints_reference[index_2]
plotDTW(query,timePoints_query,reference,
        timePoints_reference,alignment,
        ref_col="black",query_col="red",
        title="DTW")
```



Step 3: Truncation

```
index_alignableRegion=alignableRegionIndex(aligned_timePoints_query,
                                           aligned_timePoints_reference)
alignableRegion_values_query=aligned_values_query[index_alignableRegion]
alignableRegion_values_reference=aligned_values_reference[index_alignableRegion]
alignableRegion_timePoints_query=aligned_timePoints_query[index_alignableRegion]
alignableRegion_timePoints_reference=aligned_timePoints_reference[index_alignableRegion]
plotExpAlignment(query,timePoints_query,
                 reference,timePoints_reference,
                 alignment,alignableRegion_values_query,
                 alignableRegion_timePoints_query,
                 alignableRegion_values_reference,
                 alignableRegion_timePoints_reference,addAlignmentGreyLine=T,
                 title="Truncation")
```



Step 4: Calculating four measurements that jointly assess gene pair temporal similarity

1. Percentage of alignment for query – "percentageAlignmentQuery"
2. Percentage of alignment for reference – "percentageAlignmentReference"
3. Aligned gene expression correlation – "Rho"
4. Likelihood of alignment arising by chance – "pValueRho"

```
percentageAlignmentQuery=percentageAlignment(timePoints_query,
        timePoints_reference,alignableRegion_timePoints_query,
        alignableRegion_timePoints_reference)["percentage_alignment_query"]
percentageAlignmentQuery
```

```
## percentage_alignment_query
##                  0.9166667
```

```
percentageAlignmentReference=percentageAlignment(timePoints_query,
        timePoints_reference,alignableRegion_timePoints_query,
        alignableRegion_timePoints_reference)["percentage_alignment_reference"]
percentageAlignmentReference
```

```
## percentage_alignment_reference
##                      0.9082569
```

```
Rho=spearmanCorrelation(alignableRegion_values_query,
        alignableRegion_values_reference)
Rho
```

```
## rho
##   1
```

```
pValueRho=getPValueRho(Rho,
        query,timePoints_query,
        reference,timePoints_reference)
pValueRho
```

```
## [1] 5.372953e-17
```

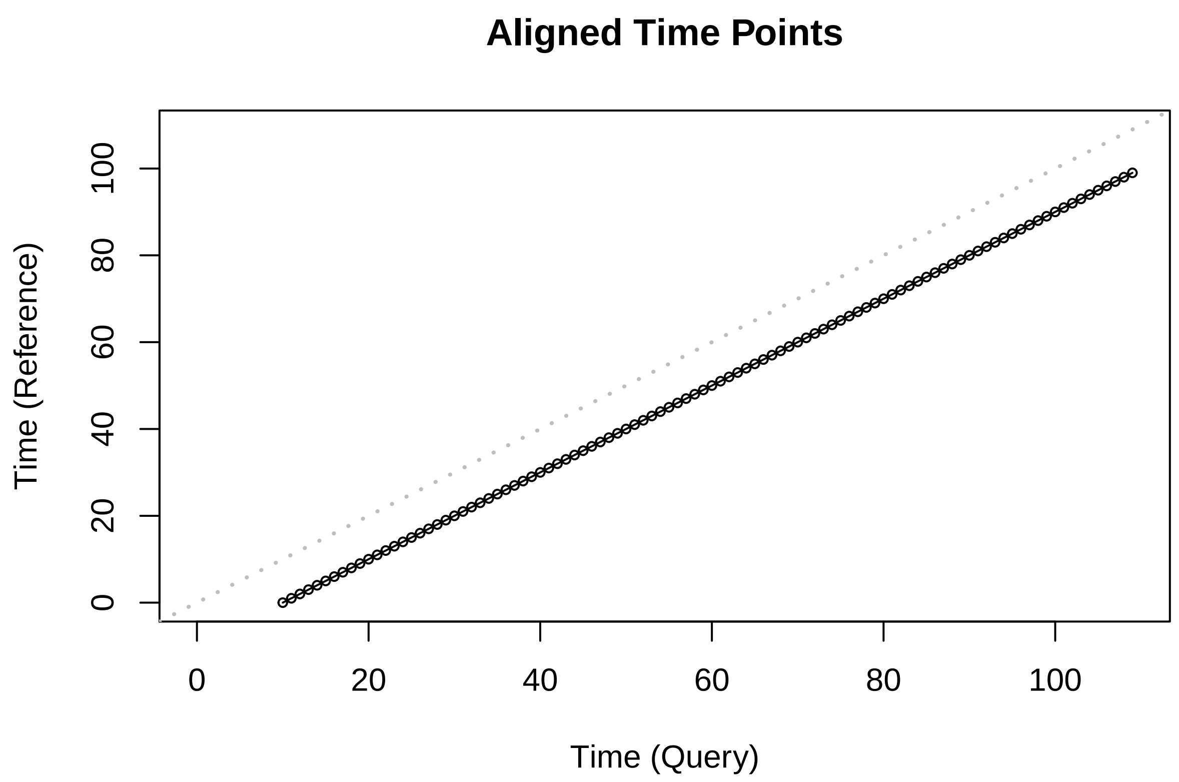
Step 5: Calculating pairwise relationship between aligned query time and aligned reference time

For each query time point within truncated alignment, TimeMeter groups and calculates the average corresponding aligned reference time.

```
alignableRegion_timePoints_query_merged=mergeReferencePoints(alignableRegion_timePoints_query,
    alignableRegion_timePoints_reference)["aligned_timePoints_query_merged"][[1]]
alignableRegion_timePoints_reference_merged=mergeReferencePoints(alignableRegion_timePoints_query,
    alignableRegion_timePoints_reference)["aligned_timePoints_reference_merged"][[1]]
segmentedRegression_out=segmentedRegression(alignableRegion_timePoints_query_merged,
                                            alignableRegion_timePoints_reference_merged)
breakPointsMatrix=fetchBreakPoints(segmentedRegression_out)
breakPointsMatrix
```

```
##      x_start x_end slope intercept
## [1,]      10   109     1       -10
```

```
plotPairwiseTimePoints(alignableRegion_timePoints_query_merged,
                       alignableRegion_timePoints_reference_merged,
                       breakPointsMatrix,
                       title="Aligned Time Points")
```



##

Step 6: Calculating progression advance score (PAS)

```
PAS=getPAS(segmentedRegression_out)$PAS
PAS
```

```
## [1] -10
```

More Examples

Dynamical Speed Difference

```
data=simdata$Different_speed_1.5
gene=data$gene
query=data$query
query
```

```
##   [1]  0.000000000  0.095007999  0.189156463  0.281593632  0.371483228
##   [6]  0.458012022  0.540397190  0.617893394  0.689799528  0.755465058
##  [11]  0.814295909  0.865759839  0.909391257  0.944795427  0.971652051
##  [16]  0.989718156  0.998830299  0.998906042  0.989944700  0.972027347
##  [21]  0.945316079  0.910052554  0.866555800  0.815219331  0.756507588
##  [26]  0.690951734  0.619144853  0.541736579  0.459427224  0.372961439
##  [31]  0.283121479  0.190720124  0.096593328  0.001592653 -0.093422430
##  [36] -0.187592323 -0.280065071 -0.370004075 -0.456595659 -0.539056431
##  [41] -0.616640368 -0.688645572 -0.754420611 -0.813370421 -0.864961683
##  [46] -0.908727652 -0.944272379 -0.971274291 -0.989489101 -0.998752022
##  [51] -0.998979251 -0.990168733 -0.972400176 -0.945834333 -0.910711543
##  [56] -0.867349563 -0.816140686 -0.757548200 -0.692102188 -0.620394741
##  [61] -0.543074594 -0.460841260 -0.374438704 -0.284648608 -0.192283301
## [66] -0.098178411 -0.003185302  0.091836624  0.186027706  0.278535799
## [71]  0.368523983  0.455178138  0.537714304  0.615385778  0.687489869
## [76]  0.753374251  0.812442869  0.864161332  0.908061743  0.943746936
## [81]  0.970894067  0.989257537  0.998671212  0.999049927  0.990390255
## [86]  0.972770540  0.946350188  0.911368222  0.868141125  0.817059971
## [91]  0.758586890  0.693250887  0.621643055  0.544411231  0.462254127
## [96]  0.375915019  0.286175015  0.193845990  0.099763245  0.004777943
```

```r
timePoints_query=data$timePoints_query
timePoints_query
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
##  [18]  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34
##  [35]  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51
##  [52]  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68
##  [69]  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85
##  [86]  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
```

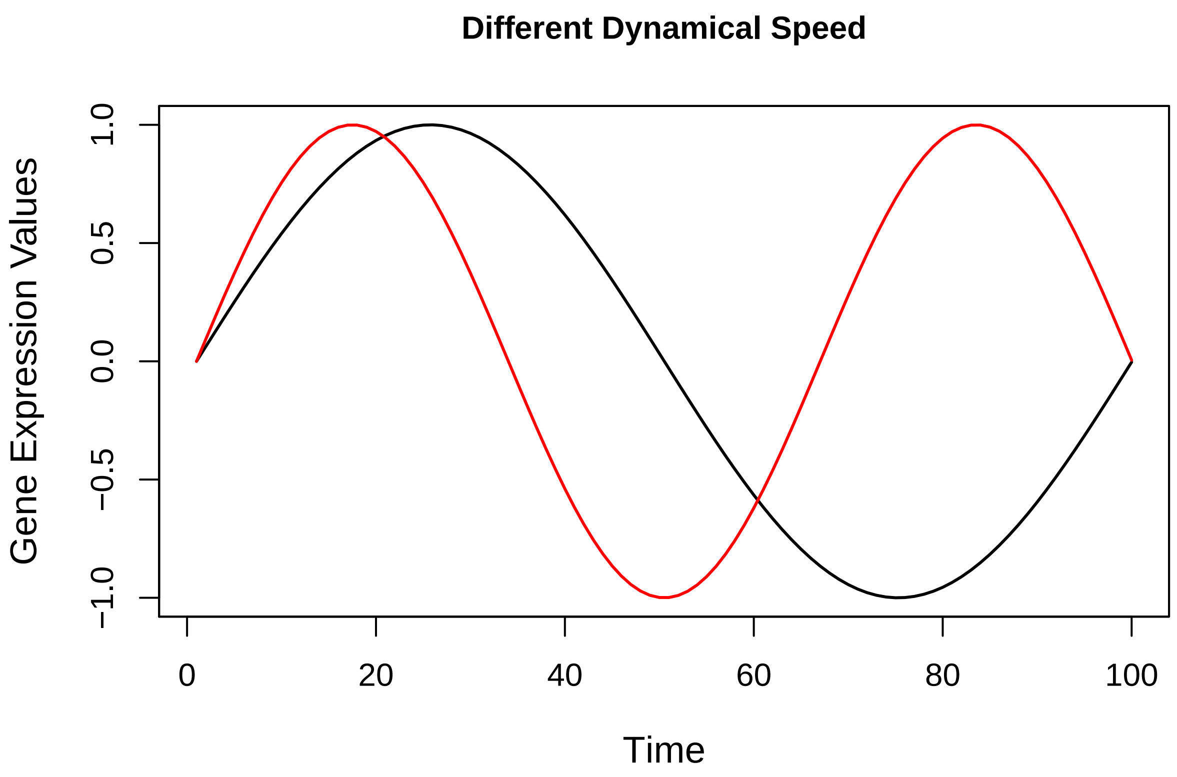
```r
reference=data$reference
reference
```

```
##   [1]  0.000000000  0.063391810  0.126528621  0.189156463  0.251023411
##   [6]  0.311880600  0.371483228  0.429591541  0.485971792  0.540397190
##  [11]  0.592648804  0.642516449  0.689799528  0.734307842  0.775862355
##  [16]  0.814295909  0.849453903  0.881194913  0.909391257  0.933929513
##  [21]  0.954710977  0.971652051  0.984684590  0.993756170  0.998830299
##  [26]  0.999886566  0.996920723  0.989944700  0.978986558  0.964090378
##  [31]  0.945316079  0.922739183  0.896450508  0.866555800  0.833175314
##  [36]  0.796443324  0.756507588  0.713528750  0.667679694  0.619144853
##  [41]  0.568119460  0.514808768  0.459427224  0.402197603  0.343350116
##  [46]  0.283121479  0.221753968  0.159494436  0.096593328  0.033303666
##  [51] -0.030119962 -0.093422430 -0.156349101 -0.218646847 -0.280065071
##  [56] -0.340356714 -0.399279251 -0.456595659 -0.512075381 -0.565495246
##  [61] -0.616640368 -0.665305014 -0.711293425 -0.754420611 -0.794513090
##  [66] -0.831409587 -0.864961683 -0.895034413 -0.921506807 -0.944272379
##  [71] -0.963239552 -0.978332030 -0.989489101 -0.996665887 -0.999833518
##  [76] -0.998979251 -0.994106524 -0.985234937 -0.972400176 -0.955653871
##  [81] -0.935063385 -0.910711543 -0.882696304 -0.851130359 -0.816140686
##  [86] -0.777868033 -0.736466353 -0.692102188 -0.644953996 -0.595211434
##  [91] -0.543074594 -0.488753200 -0.432465763 -0.374438704 -0.314905441
##  [96] -0.254105449 -0.192283301 -0.129687681 -0.066570383 -0.003185302
```

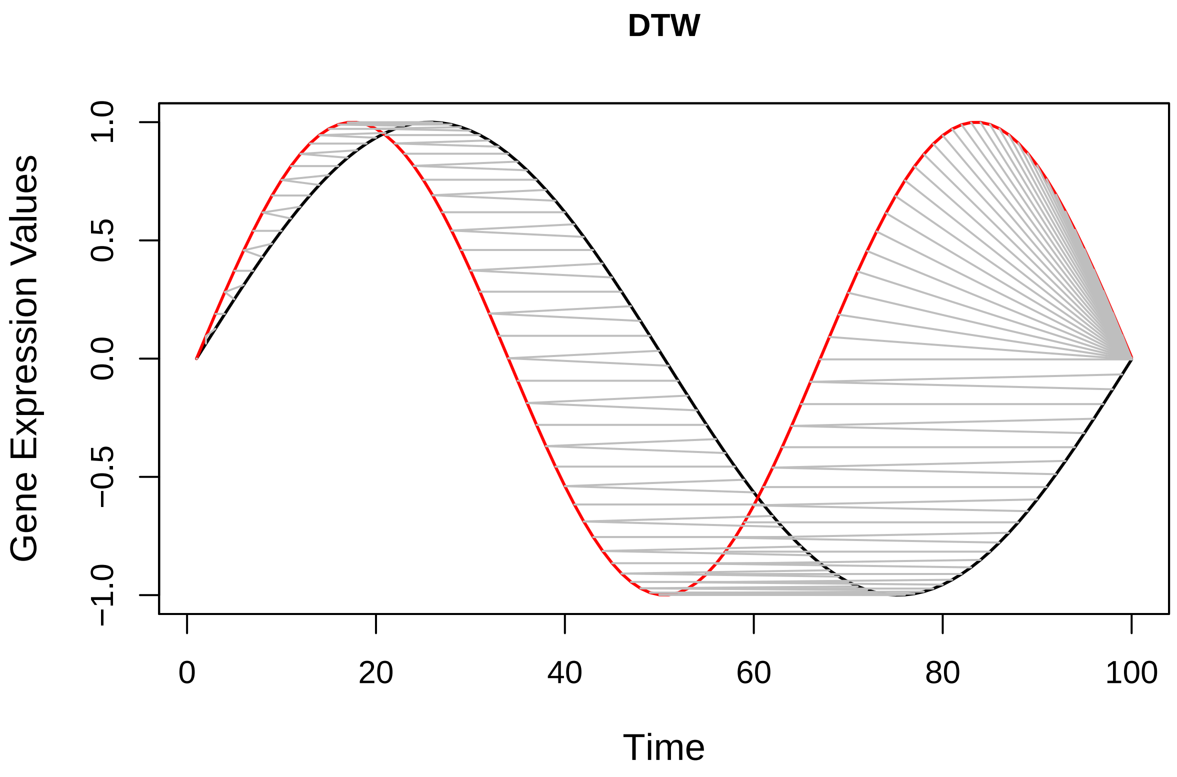
```r
timePoints_reference=data$timePoints_reference
timePoints_reference
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
##  [18]  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34
##  [35]  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51
##  [52]  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68
##  [69]  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85
##  [86]  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
```

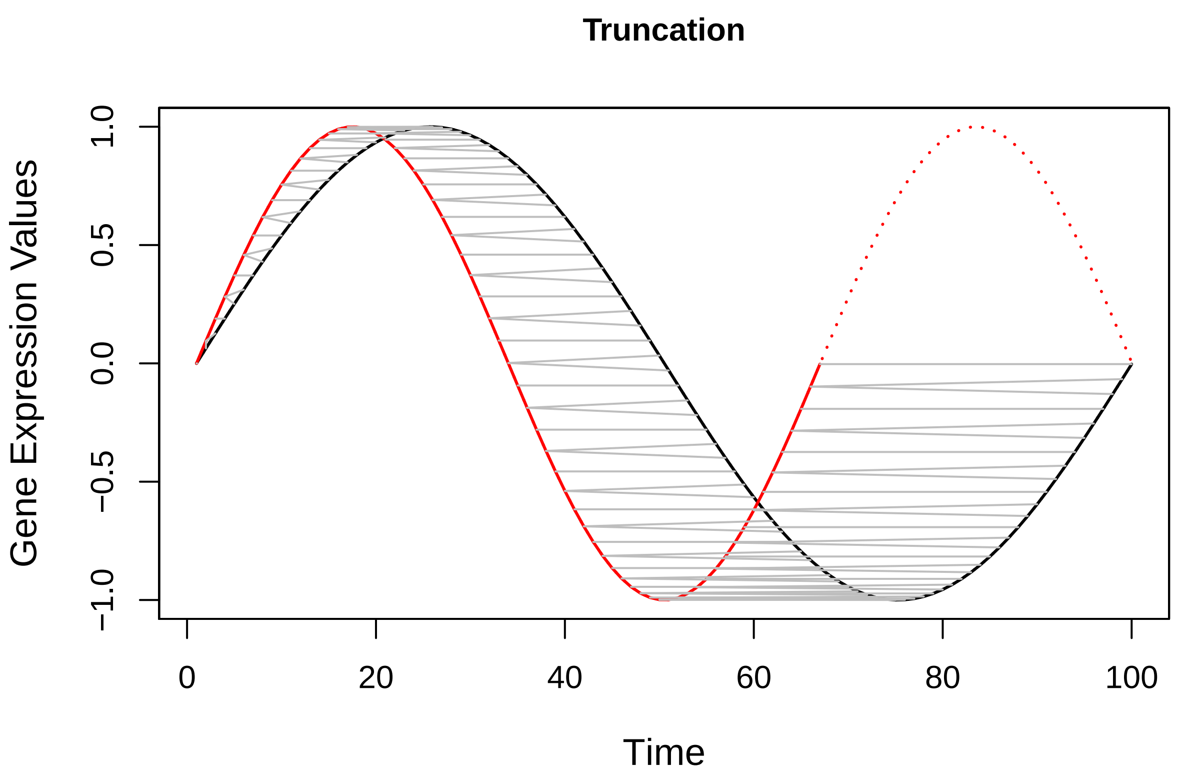
```r
plotExpOriginal(query,timePoints_query,reference,timePoints_reference,
                title="Different Dynamical Speed",ref_col="black",query_col="red")
```



```
alignment=dtw(query,reference)
dtw_results=list(alignment$index1,alignment$index2)
index_1=dtw_results[[1]]
index_2=dtw_results[[2]]
aligned_values_query=query[index_1]
aligned_values_reference=reference[index_2]
aligned_timePoints_query=timePoints_query[index_1]
aligned_timePoints_reference=timePoints_reference[index_2]
plotDTW(query,timePoints_query,reference,
        timePoints_reference,alignment,
        ref_col="black",query_col="red",
        title="DTW")
```



```
index_alignableRegion=alignableRegionIndex(aligned_timePoints_query,
                                           aligned_timePoints_reference)
alignableRegion_values_query=aligned_values_query[index_alignableRegion]
alignableRegion_values_reference=aligned_values_reference[index_alignableRegion]
alignableRegion_timePoints_query=aligned_timePoints_query[index_alignableRegion]
alignableRegion_timePoints_reference=aligned_timePoints_reference[index_alignableRegion]
plotExpAlignment(query,timePoints_query,
                 reference,timePoints_reference,
                 alignment,alignableRegion_values_query,
                 alignableRegion_timePoints_query,
                 alignableRegion_values_reference,
                 alignableRegion_timePoints_reference,addAlignmentGreyLine=T,
                 title="Truncation")
```



```
percentageAlignmentQuery=percentageAlignment(timePoints_query,
        timePoints_reference,alignableRegion_timePoints_query,
        alignableRegion_timePoints_reference)["percentage_alignment_query"]
percentageAlignmentQuery
```

```
## percentage_alignment_query
##                  0.6666667
```

```
percentageAlignmentReference=percentageAlignment(timePoints_query,
        timePoints_reference,alignableRegion_timePoints_query,
        alignableRegion_timePoints_reference)["percentage_alignment_reference"]
percentageAlignmentReference
```

```
## percentage_alignment_reference
##                              1
```

```
Rho=spearmanCorrelation(alignableRegion_values_query,
        alignableRegion_values_reference)
Rho
```

```
## rho
##   1
```

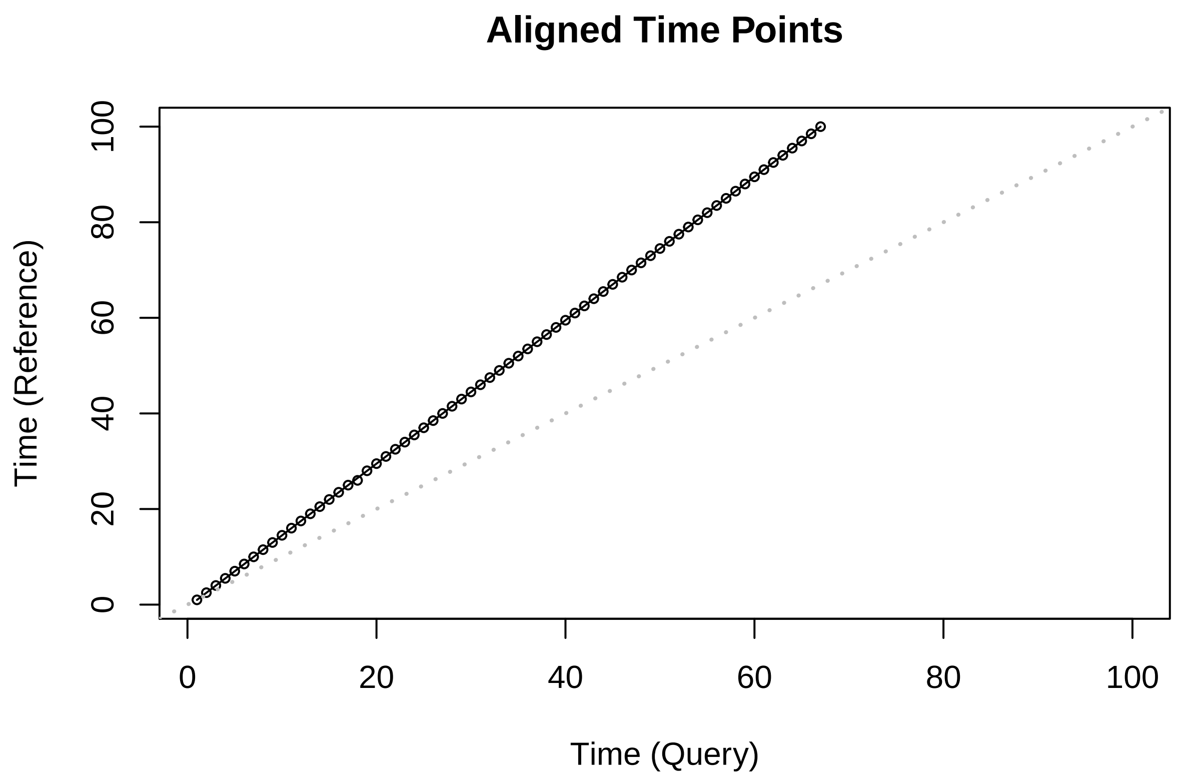
```
pValueRho=getPValueRho(Rho,
        query,timePoints_query,
        reference,timePoints_reference)
pValueRho
```

```
## [1] 2.702507e-13
```

```
alignableRegion_timePoints_query_merged=mergeReferencePoints(alignableRegion_timePoints_query,
    alignableRegion_timePoints_reference)["aligned_timePoints_query_merged"][[1]]
alignableRegion_timePoints_reference_merged=mergeReferencePoints(alignableRegion_timePoints_query,
    alignableRegion_timePoints_reference)["aligned_timePoints_reference_merged"][[1]]
segmentedRegression_out=segmentedRegression(alignableRegion_timePoints_query_merged,
                                            alignableRegion_timePoints_reference_merged)
breakPointsMatrix=fetchBreakPoints(segmentedRegression_out)
breakPointsMatrix
```

```
##      x_start x_end    slope  intercept
## [1,]       1    67 1.500319 -0.5183175
```

```
plotPairwiseTimePoints(alignableRegion_timePoints_query_merged,
                       alignableRegion_timePoints_reference_merged,
                       breakPointsMatrix,
                       title="Aligned Time Points")
```



```
PAS=getPAS(segmentedRegression_out)$PAS
PAS
```

```
## [1] 16.49254
```

Mixture patterns

```
data=simdata$Complex
gene=data$gene
query=data$query
query
```

```
##   [1]  0.000000000  0.127812689  0.253528821  0.375086232  0.490490971
##   [6]  0.597850014  0.695402314  0.781547686  0.854873057  0.914175645
##  [11]  0.958482690  0.987067406  0.999460909  0.995459905  0.975130022
##  [16]  0.938804739  0.887079913  0.820804004  0.741064158  0.649168377
##  [21]  0.546624059  0.435113276  0.316465176  0.192625987  0.065627086
##  [26] -0.062448318 -0.189499361 -0.313441980 -0.432243100 -0.543953984
##  [31] -0.646742201 -0.738921678 -0.818980363 -0.885605027 -0.937702800
##  [36] -0.974419105 -0.995151672 -0.999560417 -0.987573021 -0.959386118
##  [41] -0.915462069 -0.856521374 -0.783530858 -0.697687810 -0.600400345
##  [46] -0.493264302 -0.378037071 -0.256608766 -0.130971217 -0.003185302
##  [51]  0.000000000  0.063391810  0.126528621  0.189156463  0.251023411
##  [56]  0.311880600  0.371483228  0.429591541  0.485971792  0.540397190
##  [61]  0.592648804  0.642516449  0.689799528  0.734307842  0.775862355
##  [66]  0.814295909  0.849453903  0.881194913  0.909391257  0.933929513
##  [71]  0.954710977  0.971652051  0.984684590  0.993756170  0.998830299
##  [76]  0.999886566  0.996920723  0.989944700  0.978986558  0.964090378
##  [81]  0.945316079  0.922739183  0.896450508  0.866555800  0.833175314
##  [86]  0.796443324  0.756507588  0.713528750  0.667679694  0.619144853
##  [91]  0.568119460  0.514808768  0.459427224  0.402197603  0.343350116
##  [96]  0.283121479  0.221753968  0.159494436  0.096593328  0.033303666
## [101] -0.030119962 -0.093422430 -0.156349101 -0.218646847 -0.280065071
## [106] -0.340356714 -0.399279251 -0.456595659 -0.512075381 -0.565495246
## [111] -0.616640368 -0.665305014 -0.711293425 -0.754420611 -0.794513090
## [116] -0.831409587 -0.864961683 -0.895034413 -0.921506807 -0.944272379
## [121] -0.963239552 -0.978332030 -0.989489101 -0.996665887 -0.999833518
## [126] -0.998979251 -0.994106524 -0.985234937 -0.972400176 -0.955653871
## [131] -0.935063385 -0.910711543 -0.882696304 -0.851130359 -0.816140686
## [136] -0.777868033 -0.736466353 -0.692102188 -0.644953996 -0.595211434
## [141] -0.543074594 -0.488753200 -0.432465763 -0.374438704 -0.314905441
## [146] -0.254105449 -0.192283301 -0.129687681 -0.066570383 -0.003185302
```

```
timePoints_query=data$timePoints_query
timePoints_query
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
##  [18]  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34
##  [35]  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51
##  [52]  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68
##  [69]  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85
##  [86]  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102
## [103] 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119
## [120] 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136
## [137] 137 138 139 140 141 142 143 144 145 146 147 148 149 150
```

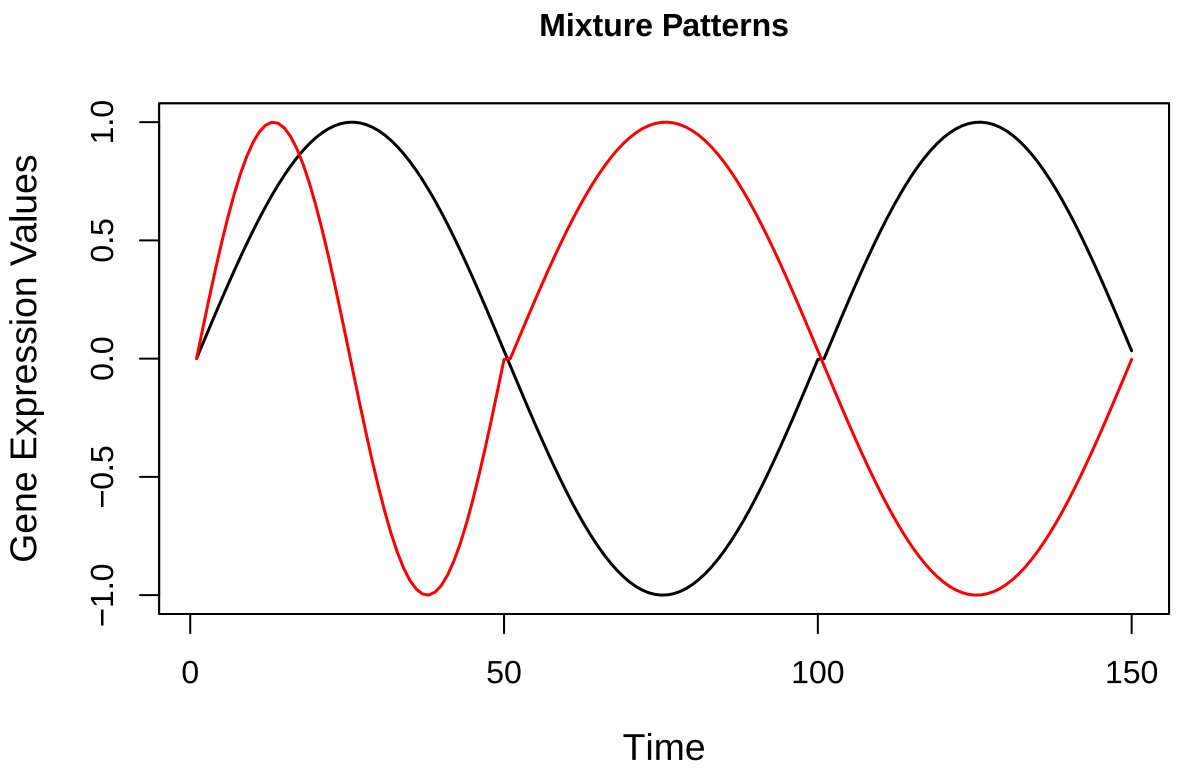
```
reference=data$reference
reference
```

```
##   [1]  0.000000000  0.063391810  0.126528621  0.189156463  0.251023411
##   [6]  0.311880600  0.371483228  0.429591541  0.485971792  0.540397190
##  [11]  0.592648804  0.642516449  0.689799528  0.734307842  0.775862355
##  [16]  0.814295909  0.849453903  0.881194913  0.909391257  0.933929513
##  [21]  0.954710977  0.971652051  0.984684590  0.993756170  0.998830299
##  [26]  0.999886566  0.996920723  0.989944700  0.978986558  0.964090378
##  [31]  0.945316079  0.922739183  0.896450508  0.866555800  0.833175314
##  [36]  0.796443324  0.756507588  0.713528750  0.667679694  0.619144853
##  [41]  0.568119460  0.514808768  0.459427224  0.402197603  0.343350116
##  [46]  0.283121479  0.221753968  0.159494436  0.096593328  0.033303666
##  [51] -0.030119962 -0.093422430 -0.156349101 -0.218646847 -0.280065071
##  [56] -0.340356714 -0.399279251 -0.456595659 -0.512075381 -0.565495246
##  [61] -0.616640368 -0.665305014 -0.711293425 -0.754420611 -0.794513090
##  [66] -0.831409587 -0.864961683 -0.895034413 -0.921506807 -0.944272379
##  [71] -0.963239552 -0.978332030 -0.989489101 -0.996665887 -0.999833518
##  [76] -0.998979251 -0.994106524 -0.985234937 -0.972400176 -0.955653871
##  [81] -0.935063385 -0.910711543 -0.882696304 -0.851130359 -0.816140686
##  [86] -0.777868033 -0.736466353 -0.692102188 -0.644953996 -0.595211434
##  [91] -0.543074594 -0.488753200 -0.432465763 -0.374438704 -0.314905441
##  [96] -0.254105449 -0.192283301 -0.129687681 -0.066570383 -0.003185302
## [101]  0.000000000  0.063391810  0.126528621  0.189156463  0.251023411
## [106]  0.311880600  0.371483228  0.429591541  0.485971792  0.540397190
## [111]  0.592648804  0.642516449  0.689799528  0.734307842  0.775862355
## [116]  0.814295909  0.849453903  0.881194913  0.909391257  0.933929513
## [121]  0.954710977  0.971652051  0.984684590  0.993756170  0.998830299
## [126]  0.999886566  0.996920723  0.989944700  0.978986558  0.964090378
## [131]  0.945316079  0.922739183  0.896450508  0.866555800  0.833175314
## [136]  0.796443324  0.756507588  0.713528750  0.667679694  0.619144853
## [141]  0.568119460  0.514808768  0.459427224  0.402197603  0.343350116
## [146]  0.283121479  0.221753968  0.159494436  0.096593328  0.033303666
```

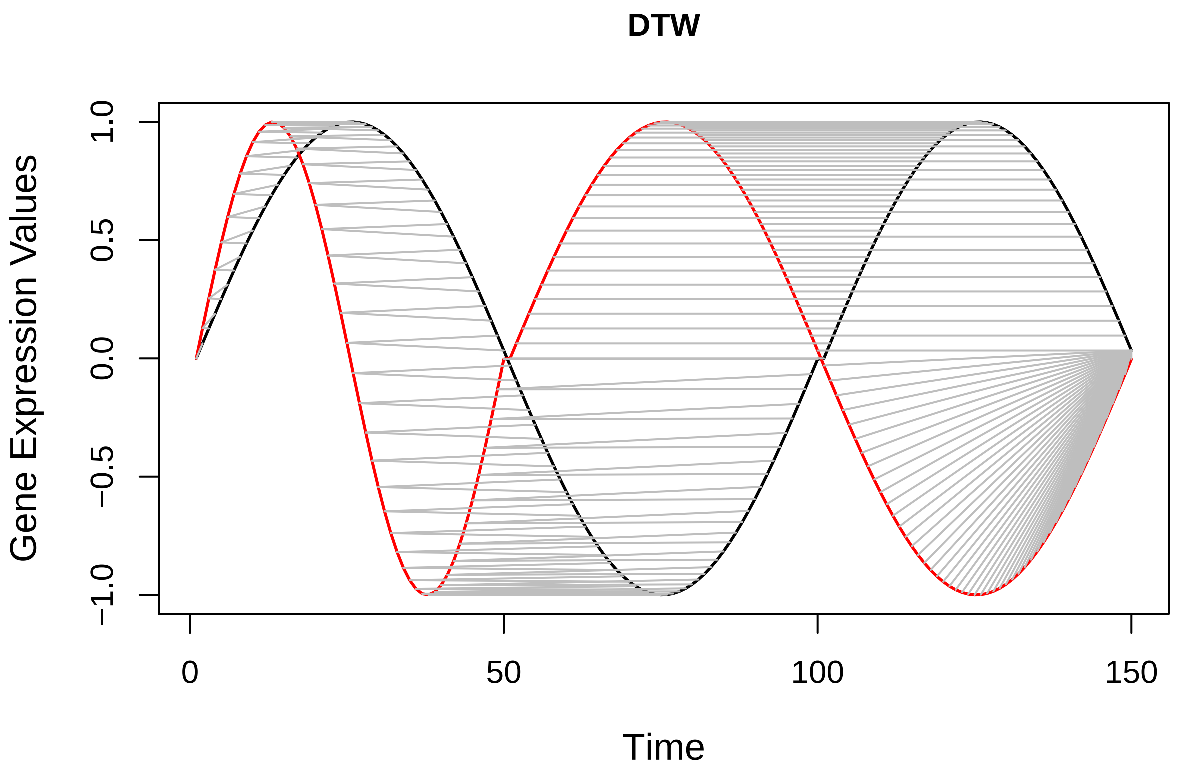
```
timePoints_reference=data$timePoints_reference
timePoints_reference
```

```
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
##  [18]  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34
##  [35]  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51
##  [52]  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68
##  [69]  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85
##  [86]  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102
## [103] 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119
## [120] 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136
## [137] 137 138 139 140 141 142 143 144 145 146 147 148 149 150
```

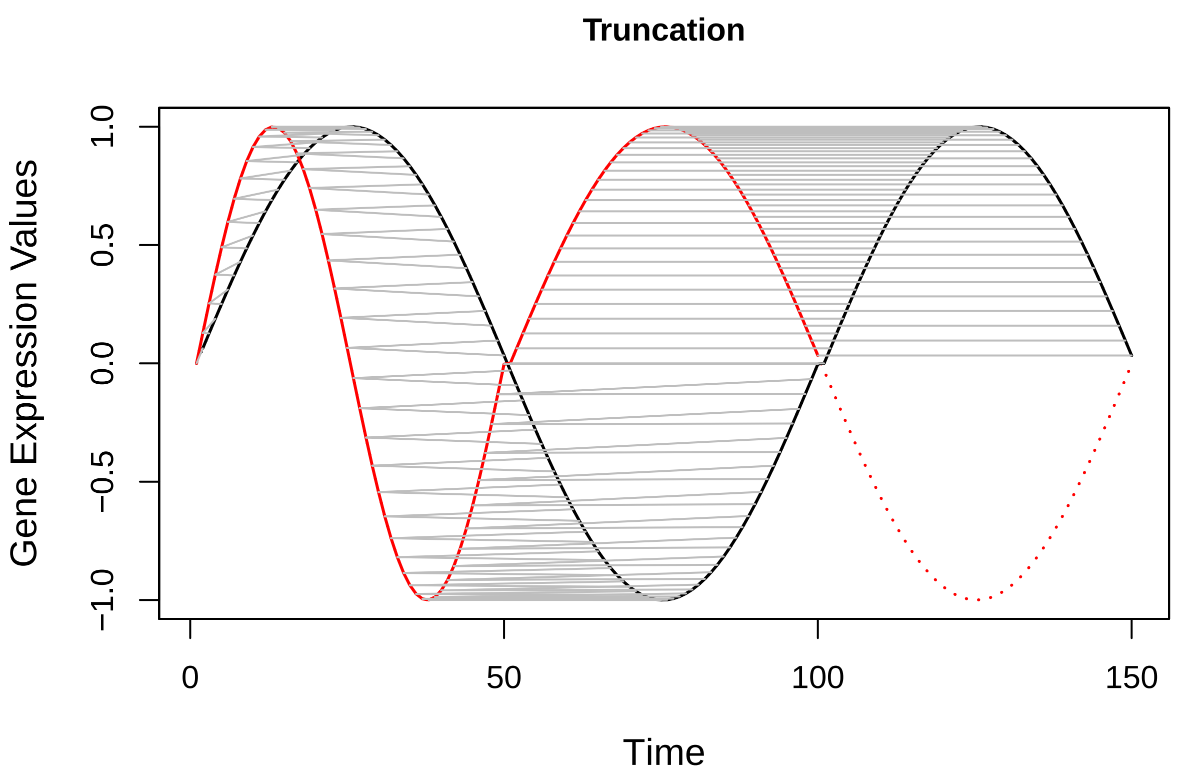
```
plotExpOriginal(query,timePoints_query,reference,timePoints_reference,
                title="Mixture Patterns",ref_col="black",query_col="red")
```



```
alignment=dtw(query,reference)
dtw_results=list(alignment$index1,alignment$index2)
index_1=dtw_results[[1]]
index_2=dtw_results[[2]]
aligned_values_query=query[index_1]
aligned_values_reference=reference[index_2]
aligned_timePoints_query=timePoints_query[index_1]
aligned_timePoints_reference=timePoints_reference[index_2]
plotDTW(query,timePoints_query,reference,
        timePoints_reference,alignment,
        ref_col="black",query_col="red",
        title="DTW")
```



```
index_alignableRegion=alignableRegionIndex(aligned_timePoints_query,
                                           aligned_timePoints_reference)
alignableRegion_values_query=aligned_values_query[index_alignableRegion]
alignableRegion_values_reference=aligned_values_reference[index_alignableRegion]
alignableRegion_timePoints_query=aligned_timePoints_query[index_alignableRegion]
alignableRegion_timePoints_reference=aligned_timePoints_reference[index_alignableRegion]
plotExpAlignment(query,timePoints_query,
                 reference,timePoints_reference,
                 alignment,alignableRegion_values_query,
                 alignableRegion_timePoints_query,
                 alignableRegion_values_reference,
                 alignableRegion_timePoints_reference,addAlignmentGreyLine=T,
                 title="Truncation")
```



```
percentageAlignmentQuery=percentageAlignment(timePoints_query,
        timePoints_reference,alignableRegion_timePoints_query,
        alignableRegion_timePoints_reference)["percentage_alignment_query"]
percentageAlignmentQuery
```

```
## percentage_alignment_query
##                  0.6644295
```

```
percentageAlignmentReference=percentageAlignment(timePoints_query,
        timePoints_reference,alignableRegion_timePoints_query,
        alignableRegion_timePoints_reference)["percentage_alignment_reference"]
percentageAlignmentReference
```

```
## percentage_alignment_reference
##                      0.9932886
```

```
Rho=spearmanCorrelation(alignableRegion_values_query,
        alignableRegion_values_reference)
Rho
```

```
## rho
##   1
```

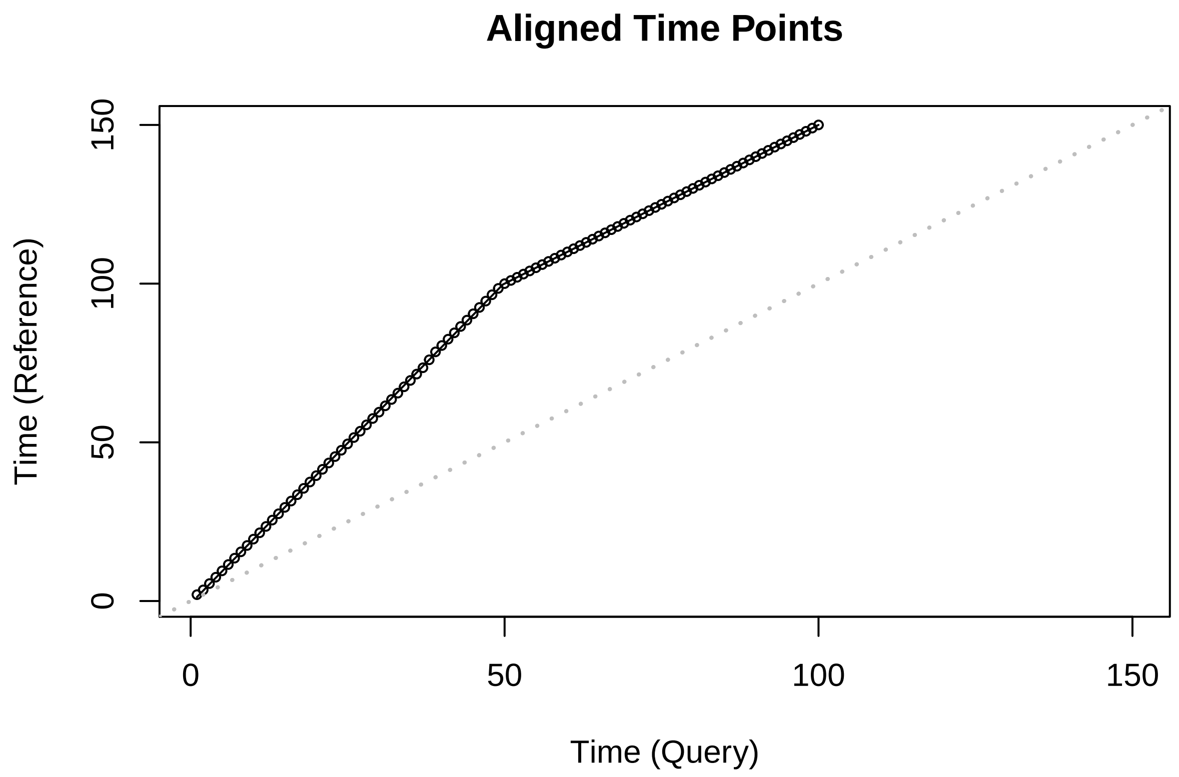
```
pValueRho=getPValueRho(Rho,
        query,timePoints_query,
        reference,timePoints_reference)
pValueRho
```

```
## [1] 5.055047e-29
```

```
alignableRegion_timePoints_query_merged=mergeReferencePoints(alignableRegion_timePoints_query,
    alignableRegion_timePoints_reference)["aligned_timePoints_query_merged"][[1]]
alignableRegion_timePoints_reference_merged=mergeReferencePoints(alignableRegion_timePoints_query,
    alignableRegion_timePoints_reference)["aligned_timePoints_reference_merged"][[1]]
segmentedRegression_out=segmentedRegression(alignableRegion_timePoints_query_merged,
                                             alignableRegion_timePoints_reference_merged)
breakPointsMatrix=fetchBreakPoints(segmentedRegression_out)
breakPointsMatrix
```

```
##      x_start  x_end    slope  intercept
## [1,]    1.00  49.74 2.020765 -0.7742347
## [2,]   49.74 100.00 1.000000 49.9986316
```

```
plotPairwiseTimePoints(alignableRegion_timePoints_query_merged,
                       alignableRegion_timePoints_reference_merged,
                       breakPointsMatrix,
                       title="Aligned Time Points")
```



```
PAS=getPAS(segmentedRegression_out)$PAS
PAS
```

```
## [1] 37.75157
```

Adjusted PAS

The progression advance score (PAS) measures the “absolute” progression difference between two similar temporal pattern (STP) genes. It is an unbiased way to compare the progression difference for species of unknown developmental paces. To investigate genes with “relative” (“unexpected”) progression difference (the differential progressions that cannot be explained by species-specific developmental paces), TimeMeter calculates a adjusted PAS that represent “relative” progression difference (adjusted by the progression difference of all STP genes).

```
data(similarHumanMouse)
alignableRegion_timePoints_query_merged_all=c()
alignableRegion_timePoints_reference_merged_all=c()

for (k in names(similarHumanMouse)) {
    alignableRegion_timePoints_query_merged_all=c(alignableRegion_timePoints_query_merged_all,similarHu
    alignableRegion_timePoints_reference_merged_all=c(alignableRegion_timePoints_reference_merged_all,s

}

Query_to_median_Reference=tapply(alignableRegion_timePoints_reference_merged_all,as.factor(alignableReg

Query_Time=as.numeric(names(Query_to_median_Reference))
Query_Time
```

```
##  [1]  0  1  2  3  4  5  6  7  8 10 12 14 16 18 20 21 22 24 26 30 32 34 36
## [24] 38 40 42
```

```
Reference_Aligned_Time_Median=as.numeric(Query_to_median_Reference)
Reference_Aligned_Time_Median
```

```
##  [1]  1.0  1.0  2.0  2.0  2.5  3.0  3.0  4.0  4.0  4.5  5.0  5.0  6.0  6.0
## [15]  6.5  7.0  7.0  7.0  8.0  9.0 10.0 12.0 14.0 15.0 17.9 18.0
```

```
PAS=similarHumanMouse[['ASIC4']]$PAS
PAS
```

```
## [1] -13.70301
```

```
PAS_adjusted=adjustedPAS(Query_Time,Reference_Aligned_Time_Median,PAS)
PAS_adjusted
```

```
## [1] -0.4188524
```

Citation

Jiang, Peng, et al. TimeMeter assesses temporal gene expression similarity and identifies differentially progressing genes. *Nucleic Acids Research* (2020).

Contact

Peng Jiang, Ph.D
Computational Biologist
Regenerative Biology Laboratory
Morgridge Institute for Research
330 N. Orchard St., Madison, WI
Tel: 608-316-4479
Email: PJiang@morgridge.org