Java Essential Dynamics (JED)

User Manual and Tutorial

Acknowledgement

Dr. Charles David wrote the JED program during his Ph.D. studies in Bioinformatics and Computational Biology at UNC Charlotte under the direction of Dr. Donald Jacobs. Partial support for this work came from NIH grants (GM073082 and HL093531), from the Center of Biomedical Engineering and Science, and the Department of Physics and Optical Science.

Citation

Charles C. David and Donald J. Jacobs. *JED: Java Essential Dynamics*. BMC (publication information).

Table of Contents:

I. Introduction 2
II. Understanding JED 4
III. Overview of Using JED 6
IV. Using JED Driver 7
A. The Preliminary Run 7
B. Debugging Crashes Part I 9
C. Performing cPCA 10
D. Performing dPCA 15
E. Performing dpPCA 20
F. Debugging Crashes Part II 24
G. Performing Cartesian Mode Visualization 25
V. Using JED Batch driver 27
A. JED Batch Driver Input File Format: The Preliminary Run 27
B. JED Batch Driver Input File Format: cPCA 28
C. JED Batch Driver Input File Format: dPCA 29
D. JED Batch Driver Input File Format: dpPCA 30
E. JED Batch Driver Input File Format: cPCA, dPCA, dpPCA, VIZ 31
F. Debugging Batch Crashes 31
VI. Additional Types of Analysis 32
A. Pooling Data 32
B. Subspace Analysis 33
Appendix 35
I. Input File Formats 35
II. Output File Formats 37

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

I. INTRODUCTION

Java Essential Dynamics (JED) is a java library (a package of programs) for analyzing protein trajectories. The trajectories may be derived from MD, FIRST/FRODA, or any other dynamic simulations that output a trajectory as a set of PDB files. The program can handle single chain PDB files with no chain identifier as well as multi chain PDB files that use chain IDs. The user may specify the set of residues to be considered for the analysis, and this set need not be contiguous. A variety of utility tools are provided that use **Principal Component Analysis** (PCA) that are not found in MD-simulation packages or other stand alone PCA software, especially in regards to comparative analysis of multiple trajectories. JED is capable of running on any platform with a suitable Java Runtime Environment (JRE).

Expected Input to JED:

Ideally, each PDB structure must follow standard PDB-format. Note that some deviations from standard will often work fine, but JED expects standard format. Moreover, it is required that the structures have been prepared in such a way that there are no gaps in residue labeling. If residues or consecutive regions of residues are missing, these need to be fixed, or the residue labeling has to be altered in order to remove gaps in residue labels. The first residue label must start at 1 or higher. No 0 or negative numbers are allowed for residue labels. All preprocessing of the PDB files must be done with external software before using JED. It is convenient to label PDB files using leading zeros in the name of the files to simplify tracking time progression. For example, if a simulation generates 100,000 frames in the trajectory, it is best to name the PDB files like <file_name_000000>, <file_name_000001>, … <file_name_100001>, which specifies that relative to the starting structure 100,000 frames are generated in successive order. Although this naming scheme is not required it is highly recommended because it allows the user to track time order easily on operating systems that sort order by literal alphabetic-characters, rather than interpreting 34 is less than 100, for example.

JED Preprocessing Output:

As a pre-processing step, JED reads in all PDB files in a specified directory and aligns all the structures in the trajectory to a specified reference structure using a quaternion alignment algorithm. A matrix of the **read PDB coordinates,** obtained from all the residues in the input PDB files, is created so that it can be used for all subsequent JED runs. A list of all the residues (**residue list**) found in the PDB files (along with the chain IDs when appropriate) is generated. The original and transformed **conformation rmsd** are determined for each member structure in the trajectory relative to the specified reference structure. The **residue rmsd** (also commonly referred to as **RMSF**) is determined from the entire trajectory. An **edited PDB file** is also generated where the B-factors are replaced with the **residue rmsd** values for visualization purposes. This output automatically happens and is non-optional.

Carbon Alpha Atoms:

The current implementation of JED only considers Cα atoms. As such, we speak about residues because the information is tied to the Cα atoms, which represents the dynamics of the residue at a coarse grained level of description. For example, the distance between two residues is modeled in JED as the distance between the two Cα atoms associated with the two residues. This choice of working only with Cα atoms allows the labeling of the Cα atoms to be associated with residue labels. For a single chain protein, this is a simple 1 to 1 mapping. For multiple chain proteins, JED also tracks the chain ID.

Different Types of PCA:

The core element of essential dynamics is to perform PCA. JED implements two variations of PCA. The first and most common method is based on Cartesian coordinates (**cPCA**). Note that cPCA using ***n* residues** will yield eigenvectors having ***3n*** components, each corresponding to one Cartesian coordinate. The second method is based on internal coordinates using either all-to-all residue distances (**dPCA**) and/or residue-pair distances (**dpPCA**). The dPCA is much more computationally expensive and the interpretation can be difficult when the number of residues in the subset exceeds ten. Note that dPCA using ***n* residues** will yield eigenvectors having ***n*(n-1)/2** components, each corresponding to one set of inter-residue distances. Thus, for five residues, one obtains ten pairs of interactions, and this becomes

difficult to interpret. In contrast, dpPCA using ***n* residue-pairs** will yield eigenvectors having ***n*** components, each corresponding to one of inter-residue distance pairs.

All of the PCA methods are performed using a **covariance matrix** (**Q**) and a **correlation matrix** (**R**). The correlation matrix is a normalized version of the covariance matrix. The results obtained from **Q** and **R** generally differ somewhat due to the inherent statistical biases in each approach. The current implementation does both and compares the results.

Conditioning of the Q and R Matrices:

JED handles the removal of outliers prior to the PCA analyses with two approaches. First, the user can specify the **percent** (a decimal [0,1]) of the structures to be removed based on the conformation rmsd. The most deviant structures are tagged as outliers and subsequently removed prior to the PCA analysis. In this first method, frames that are identified as an outlier are thrown out from the sample. Second, the user can specify a **z-score cutoff** (a decimal ≥ 0) such that when the value of a PCA variable (either a Cartesian or internal distance coordinate) has a |deviation| from the variable mean that exceeds the z-score cutoff, it is identified as an outlier. For each PCA-entry that is identified as an outlier, it is replaced with its mean. This process is done per variable over all frames, and each PCA-entry is treated independently. In this second method, a frame is never thrown away, but some entries within a frame may be modified. Both methods are intended to reduce the condition numbers of **Q** and **R**. While the first method of conditioning is most commonly employed in the protein field (if at all), the second method of conditioning is most commonly used in the field of statistics, and is the preferred method due to its superior effectiveness. It should be noted here that without applying the conditioning, the results of a PCA can be highly skewed due to the presence of outliers as such analyses are highly dependent on the quality of the sampling. It is strongly recommended to use the z-score cutoff conditioning method in all applications to avoid misinterpreting the PCA results.

Visualization of cPCA modes:

JED computes the **PCA modes** (RMSD and MSD, with and without weight by the corresponding eigenvalue) from the Cartesian eigenvectors so that they may be mapped to the residue set. As noted above, sets of structures can be generated to visually inspect the cPCA modes. Eigenvectors from dPCA cannot be mapped to the residue set in any simple way, so no mapping or visualization is attempted. The user can specify the number of Cartesian modes to visualize. Mode visualization is done by creating a set of 20 PDB files that capture the displacement of the alpha carbons for the given mode. A scale-factor parameter is adjustable to control the amount of displacement in the modes. A Pymol© script is generated to animate the frames.

Dimension Reduction Level:

The primary purpose of applying PCA to capture the essential dynamics of a protein is to reduce the large dimension of variables to a much smaller number of variables that captures the greatest variance in protein motion. The **Q** and **R** matrices, once diagonalized, provide a set of eigenvalues and eigenvectors. The eigenvalues for proteins typically fall off fast for the first several modes, out of possibly thousands of modes. The number of dimensions needed to provide a fair assessment of the essential dynamics in a protein is system-dependent. The user can specify any number (say 20, which typically is more than needed) to obtain results for all possible selections, ranging from 1 up to the maximum value that is selected. In this way, the user can see how the added dimensions help glean more information, albeit making it harder to interpret the greater number of dimensions. Eventually, the user must decide, based on their purpose/goals, the optimal number of dimensions to use for representing the essential dynamics.

Displacement Vectors:

A set of **displacement vectors** (**DV**s) based on the full conformational space is calculated using a specified reference structure. Those **DV**s are then projected onto a set of eigenvector directions to create delta vector projections (**DVP**s), which are similar to principle components (**PC**s). The **PC**s are delta vector projections, but according to the standard definition used in statistics, they are always relative to the mean conformation position as defined in the construction of the **Q** or **R** matrix. In studying the essential dynamics of a protein, it is common to use a reference structure that has a

particular physical or biochemical meaning, which is why we call these displacements **DVP**s, and not **PC**s. The DVPs are very useful to have for visualizing protein motions. For example, if the first two eigenvector directions are selected (those eigenvectors associated with the highest and second highest eigenvalues, or variance) the **DVP**s can be plotted for each frame to construct the trajectory in conformational space projected onto a two dimensional cross-section. Other eigenvector directions can be specified, allowing the user to investigate how the trajectory projects into the space defined by each eigenvector. The **DVP**s are given using un-normalized and normalized inner products, as well as weighted by the corresponding eigenvalue. The different methods highlight the structure of the data and provide scaling for visualization.

Post PCA Comparative Subspace Analysis:

JED performs a subspace analysis (**SSA**) on the two equidimensional sets of eigenvectors generated from the **Q** and **R** variants of PCA. The results provide a detailed comparison for the chosen subspace as well as comparisons for all subspace dimensions up to the dimension chosen by the user (when selecting the number of Cartesian or Distance modes to process) in an iterative fashion. This allows one to quantitatively determine how different the PCA results are due only to the choice of PCA model, while also assessing the size of the essential subspace. Additional analysis can be done using the driver programs for the Subspace Analysis class. To perform these kinds of tests, it is a best practice to first generate equidimensional sets of eigenvectors from each trajectory of interest, as well as from a pooled trajectory to use as a reference set, while ensuring that the subsets of residues analyzed are identical. Subspace analysis is done by comparing the sets of eigenvectors, directly or iteratively, and determining the root mean square inner products (**RMSIP**s), Principal Angles (**PA**s), cumulative overlap (**CO**s), cosine products (**CP**s), vectorial angular sum (**VAS**), and the maximum angle between subspaces of the given vector space.

```
(In this tutorial, code, file paths, and text file content are shown in dark blue 9 point Consolas)
```

II. Understanding JED

JED Install Instructions:

Java is **platform independent** and JREs exist for all common architectures. The machine on which JED is to be run should have **JRE version 1.6 or higher** installed. The programs can be run from compiled source or from the provided executable jar files. While JED can be installed in any directory that is part of your Java classpath, the sources must be compiled on the local machine to insure runtime integrity. When compiling from source, be sure to compile the JAMA MATRIX package as JED uses that library. Alternatively, no source code or compilation is needed to run the executable jar files. These can be placed in any directory that is on the Java classpath. For most applications, a **64bit OS is required** to address the amount of memory needed for the analyses. It is critical that the environment variable Java **CLASSPATH** be correctly set to run Java programs at the command prompt. Alternatively, you can always add the **-cp** option to the **java** command, which allows you to specify the path that contains your Java classes.

Expected Memory Requirements:

On high performance computer clusters make sure the 64 bit JRE is installed. Memory use is demanding because JED loads the complete covariance matrix (among other data structures) and performs a full matrix diagonalization, which scales as $O(N^3)$. Typically 8 to 32 GB of RAM will be needed depending on the size of the protein. For very large proteins consisting of thousands of residues and/or many tens of thousands of frames, make available as much memory per node as possible (~ 1TB). On most platforms, Java can be optimized by specifying parameters at runtime for heap space, perm space, etc.

Two Kinds of JED Drivers:

There are two driver programs for JED: One (**JED_Driver**) runs a single job using parameters specified in the input file, and the other (**JED_Batch_Driver**) runs a batch of jobs sequentially. The first is suited for running a single job at the command line or when using submit scripts on computer cluster resources. This can be implemented using job arrays so

that your jobs run in parallel rather than sequentially. The second is suited for running multiple jobs on a single computer so that a user can submit a batch of jobs, and come back a few hours later with many different jobs finished without having to launch each one separately. Note: The input file formats for the two driver programs are NOT equivalent.

Input File and Data for JED Driver:

JED requires an input file for job parameters. The format of this file will be described below. The run command takes only one argument, which is the name of the input file that includes the absolute path to the file. If no argument is specified, then JED assumes that the default input file name is used and the file is located in the same directory from which the Java Virtual Machine (JVM) was called. The default input file names are:

JED_Driver.txt for JED_Driver.java (or .jar file)
JED_Batch_Driver.txt for JED_Batch_Driver.java (or .jar file)

Each job should be assigned to its own directory, which must contain either the **PDB files** to read (for Pre-Processing runs) or the **Coordinates Matrix** to process (for all PCAs), along with the **reference PDB file** and **residue lists** for specifying the subsets of interest: Cartesian, Distance, and/or Distance Pairs.

JED Command Line format:

To run **JED_Driver** at the command prompt or within a PBS script, you can use one of the following commands:

```
java -d64 JED_Driver "/path/to/your/input/file.txt" (runs the compiled java program)
java -jar -d64 JED_Driver.jar "/path/to/your/input/file.txt" (runs the executable jar file)
```

To run **JED_Batch_Driver** at a command prompt or in a PBS script, you can use one of the following commands:

```
java -d64 JED_Batch_Driver "/path/to/your/input/file.txt")
java -jar -d64 JED_Batch_Driver.jar "/path/to/your/input/file.txt"
```

Remember to include command line switches to optimize the Java runtime environment for your jobs.

Organization of Output Files:

Output files from JED are written to subdirectories within the working directory, structured to organize the multitude of files produced in a meaningful manner. The top level of this directory tree is named "JED_RESULTS_**$description**", where **$description** is a user set parameter that succinctly describes the job. Limbs of the tree separate Cartesian PCA (**cPCA**), Distance PCA (**dPCA**), Distance-Pair PCA (**dpPCA**), and Mode Visualization Analysis (**VIZ**), when present. Each of these in turn contains limbs for **Q** (**COV**) and **R** (**CORR**) compartmentalization. The PCA directories also contains a subdirectory for the subspace analysis (**SSA**). Most of the output file names include the **number of residues or pairs** in the selected subset for reference plus a description of the file contents.

Current Limitations:

Initial input of the protein trajectory must be done using PDB files that are expected to conform to the standard format, or a matrix of PDB coordinates containing the alpha carbon atomic positions only (see below for a description of this file). Only carbon-alpha atomic positions are used to create a **Q** or **R** matrix for essential dynamic analysis.

Each PDB file must have the exact same number of residues. The matrix of alpha carbon coordinates is determined from the first PDB file read. If other files in the working directory do not match exactly, then the array sizes will not match and the program will crash. *IF JED crashes during the reading of PDBs, this is probably the reason*.

While JED can process a PDB file with missing residues and various numbering schemes, it can NOT interpret files that have alternate conformations within a given frame based on fractional **occupancy** values. Only a single conformation per

frame is allowed. Note that the original residue coordinates in the PDB files are mapped to the rows of the coordinates matrix and thus care must be exercised when specifying residue subsets.

III. Overview of Using JED

A **Preliminary Run** with NO PCA must be performed to generate the JED formatted coordinate matrix file for all the alpha carbons in the PDB files. This makes subsequent subset analyses much faster to perform. It also serves to guarantee that the specified residues for subset selection are correctly chosen. After this initialization step, the PDB files can be deleted or archived, with the exception of the reference PDB file. Once the coordinate matrix is created, it should be used for all subsequent analyses using different residue subsets and different job parameters.

The name of the coordinate file matrix produced from the PDB files is: `"original_PDB_coordinates.txt"`
The matrix packing is as follows:
Rows are coordinate variables and columns are frames.
For N residues, there are 3N rows: N x-coordinates, N y-coordinates, and N-z coordinates, stacked in that order.

- **The file to use in all subsequent JED analyses is the original_PDB_coordinates matrix.**

  This matrix contains all the residues in the PDB files and thus can be used for any subset of those residues. When a subset is chosen, a new correspondence set is generated and a new transformation is done to optimize the alignment of the structures. This removes overall translation and rotation.

In subsequent analyses, it is critical that no residues are requested that do not actually exist in the PDB files!
JED maps the specified residue list to an internal list that is aligned to the rows of the coordinates matrix. JED generates a residue list file for all residues it finds in the PDB files that it reads. This file should be edited with care when specifying residue subsets.

Note: The most critical step when using JED is in the creation of the input file. The input file must have the correct format (shown in examples below) and the entries must be accurate. If either of these conditions is violated, the program will crash, or worse, the results will be corrupt. JED provides abundant error feedback during most crashes so that the problems can be addressed.

Common Causes for JED to Crash

- If **any directory** cannot be found or if **any file** cannot be found, JED will crash.
- If unexpected format is found in **any** of the input files, JED will crash.

Note that the JED driver programs employ many checks during the reading of the input files and the execution of the program. There are checks to validate the number formats of numeric data. There are checks to ensure that enough parameters were specified for the particular job. There are checks to ensure that the number of modes requested does not exceed the actual number of modes available. JED also verifies that directories and files exist before performing any analysis. The developers have gone to great lengths to provide meaningful information when the program crashes to facilitate making the necessary corrections. The specified input file is echoed to standard out as are the assignment of parameters, while any detected problems or errors have their messages directed to standard error. In the case that a Java runtime exception is thrown, a stack trace will also be sent to standard error.

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

IV. Using JED DRIVER:

A. The Preliminary Run

i. Run Command:

```
java -jar -d 64 JED_Driver.jar "/path/JED_Driver.txt"
```

The PDB files (including the PDB reference file) must be in the working directory.
JED input file may be in the working directory.

This pre-processing step will read all PDB files in the working directory, but will perform **no PCA.**
The purpose of **no PCA** this is to generate the matrix of coordinates for performing subset analyses efficiently.

ii. Root Output Files:

These are written to the **root** of the JED Results directory tree:
`/working/directory/JED_Results_Description/`

JED LOG providing a summary of the job parameters and results:

`JED_Log.txt`

PDB READ LOG listing all the PDB files read, in the order they were read:

`PDB_READ_Log.txt`

coordinates matrix from all the alpha carbon coordinates in the PDB files:

`original_PDB_coordinates.txt`

transformed coordinates matrix, which aligns all the frames to the reference frame :

`ss_$num_res_transformed_PDB_coordinates.txt`

list of all residues found in the PDB files for subsequent editing and use:

`All_PDB_Residues_JED.txt (for Single Chain PDBs)`

`All_PDB_Residues_Multi_JED.txt (for Multi Chain PDBs)`

original and transformed conformation RMSDs:

`ss_$num_res_original_conformation_rmsds.txt`

`ss_$num_res_conformation_rmsds.txt`

residue RMSDs (RMSF):

`ss_$num_res_residue_rmsd.txt`

edited PDB file containing all the residues with the RMSF replacing B-factors:

`ss_$num_res_edidited.PDB`

coordinates z-score matrix for all the alpha carbon coordinates in the PDB files:

`ss_$num_res_coordinate_Z_scores.txt`

percent of the frames to remove based on conformation RMSD (OPTIONAL)

z-cutoff for adjusting coordinate outliers to their mean values (OPTIONAL)

If **percent** or **z-cutoff** are set to a value other than **zero**, additional output files are generated with the results.

Note: The optimal time to handle outliers is during runs that actually perform PCA and it is recommended that for pre-processing runs, these be set to zero unless one wants to inspect their data.

iii. JED Driver Input File Format: The Preliminary Run

```
---------------------------------------------------------------------------------------------------
0.00
0.00
1     $multi
0     0     0     0
/working/directory/job/
Description     reference_PDB_file.pdb
---------------------------------------------------------------------------------------------------
```

Notes: This is a **whitespace** separated file with **6 lines**.

Line 1 Field 1: specifies the **percent** (double): 0 → No frames removed

Line 2 Field 1: specifies the **z-cutoff** (double): 0 → No outliers adjusted

Line 3 Field 1 specifies **read flag**, whether to **read PDB files** (0 or 1) → 1 = yes
Field 2: specifies **multi flag**, if the PDB files are **Multi Chain** (0 or 1)

Line 4 Field 1: specifies the **number of cPCA modes** (integer) → 0 = none
Field 2: specifies the **number of dPCA modes** (integer) → 0 = none
Field 3: specifies the **number of dpPCA modes** (integer) → 0 = none
Field 4: specifies the **number of cPCA modes to Visualize**(integer) → 0 = none

Line 5 Field 1: specifies the **working directory** (String)

Line 6 Field 1: specifies the **description** (String) for the requested job
Field 2: specifies the **reference PDB** (String) for the requested job

---

Key Points:

- The **percent** can be set to check the most deviant frames in the trajectory by conformation RMSD
- The **z-cutoff** can be set to check the number of outliers per variable and examine the variable z-scores
- The **Read** flag **MUST** be set to **1** or **"read"**
  - When the **Read** flag is set to **1**, all PCAs are turned off
- The **Multi** flag must be set to **0** for Single Chain PDBs with no Chain IDs
- The **Multi** flag must be set to **1** or "multi" for Multi Chain PDBs with Chain IDs
  - **Multi Chain PDBs must have unique chain identifiers for every chain**
  - **Missing chain identifiers will cause JED to crash**

- **The file to use in all subsequent JED analyses is the original_PDB_coordinates matrix**

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

B. Debugging Crashes Part I:

Things that will generally make your life miserable…

i. Simple mistakes:

Does the path to the input file exist?
Does the input file exist in the proper location?
Does the input-file start on the first line?

Is the **number format** correct? (20.0 will NOT parse as an integer)
Are the **Read** and **Multi** flags set correctly?
Did you forget a parameter declaration?

Does the working directory string end in "**/**" or "****"?
Does the working directory exist?
Does the working directory contain PDB files of different sizes?
Does the working directory contain the reference PDB file?

Does the reference PDB file exist?
Does the reference PDB file correspond to the trajectory?

ii. More subtle mistakes:

The directory contains PDB files in non-standard format.

The directory contains PDB files with fractional occupancy data.

The directory contains PDB files with 2 chains, but no chain IDs.

The directory contains PDB files with missing chain IDs.

If the PDB file names are sorted in a different order than how they were generated, then the conformation RMSD results will not reflect what actually occurred in the simulation.

Naming the PDB files appropriately by padding the numbers with leading zeros will ensure proper sorting to prevent this problem caused by the operating system.

If the conformation RMSD is very different from what you expect, then you may be using PDB files that contain occupancy information. JED does not use that information. Your results will not be accurate.

If you have pooled data, make sure the combined matrix is constructed the way you think it is and the reference column is the frame you think it is.

C. Performing cPCA

i. Run command:

```
java -jar -d64 JED_Driver.jar "/path/JED_Driver.txt"
```

The working directory must contain: The coordinates matrix, the PDB reference file, and the residue list.
JED input file may be in the working directory.

The purpose of this type of run is to perform Essential Dynamics using cPCA based on **Q** and **R**. The user specifies the subset of interest for the analysis, which may be the entire protein or a sub-region, which can be non-contiguous, by providing a residue list file. This task is simplified since JED has already created a list of all the residues in the protein. The user can simply edit this file. The cPCA results are written to the sub-directory "cPCA" and the Visualizations of the top modes (when selected) are written to the subdirectory "VIZ". The directory cPCA has sub directories for the **Q** and **R** analysis, as does the VIZ directory.

ii. Root Output Files:

These are written to the **root** of the JED Results directory tree:
`/working/directory/JED_Results_Description/`

JED LOG providing a summary of the job parameters and results:

`JED_Log.txt`

iii. cPCA Output Files:

These are written to the **/cPCA subdirectory** of the JED Results directory tree:
`/working/directory/JED_Results_Description/cPCA/`

subset transformed coordinates matrix:

`ss_$num_res_transformed_PDB_coordinates.txt`

original and transformed conformation RMSDs:

`ss_$num_res_original_conformation_rmsds.txt`

`ss_$num_res_conformation_rmsds.txt`

residue RMSDs (RMSF):

`ss_$num_res_residue_rmsd.txt`

subset edited PDB file with the RMSF replacing B-factors:

`ss_$num_res_edited.PDB`

subset coordinates Z-Score matrix:

`ss_$num_res_coordinate_Z_scores.txt`

list of frames removed, based on the **percent** parameter:

`ss_$num_res_ Removed_Conformation_Outliers.txt`

trimmed transformed coordinate matrix:

`ss_$num_res_ trimmed_$percent_percent_PDB_coordinates_COLS.txt`

list of coordinate variables adjusted, based on the **z-cutoff** parameter:

`ss_$num_res_ adjustments_per_variable.txt`

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

adjusted transformed coordinate matrix:

`ss_$Z_threshold_$z-cutoff_adjusted_PDB_coordinates_ROWS.txt`

centroids (means) of the variables:

`ss_$num_res_centroids_of_variables.txt`

standard deviations of the variables:

`ss_$num_res_std_devs_of_centered_variables.txt`

displacement vectors:

`ss_$num_res_delta_vectors.txt`

- **The Q output files are written to the /COV subdirectory of /cPCA (shown below)**

`/working/directory/JED_Results_Description/cPCA/COV/`

- **The R output files are written to the /CORR subdirectory of /cPCA**

`/working/directory/JED_Results_Description/cPCA/CORR/`

covariance matrix:

`ss_$num_res_COV_matrix.txt`

eigenvalues:

`ss_$num_res_eigenvalues_COV.txt`

top eigenvalues:

`ss_$num_res_top_$num_of_cart_modes_eigenvalues_COV.txt`

top eigenvectors:

`ss_$num_res_top_$num_of_cart_modes_eigenvectors_COV.txt`

top pca modes and top weighted pca modes:

`ss_$num_res_top_$num_of_cart_modes_pca_modes_COV.txt`
`ss_$num_res_top_$num_of_cart_modes_weighted_pca_modes_COV.txt`

top square pca modes and top weighted square pca modes:

`ss_$num_res_top_$num_of_cart_modes_square_pca_modes_COV.txt`
`ss_$num_res_top_$num_of_cart_modes_weighted_square_pca_modes_COV.txt`

top PCs and top weighted PCs:

`ss_$num_res_top_$num_of_cart_modes_PCs_COV.txt`
`ss_$num_res_top_$num_of_cart_modes_weighted_PCs_COV.txt`

top PCs and top weighted PCs:

`ss_$num_res_top_$num_of_cart_modes_normed_PCs_COV.txt`
`ss_$num_res_top_$num_of_cart_modes_weighted_normed_PCs_COV.txt`

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

iv. <u>SSA Output Files:</u>

These are written to the **/SSA subdirectory** of /cPCA:
`/working/directory/JED_Results_Description/cPCA/SSA/`

The Fast SSA Iterated Log:

`JED_FSSA_Iterated_log.txt`

The SSA Log:

`JED_SSA_dim_$top_num_cart_modes_log.txt`

The Random SSA Log

`JED_Random_SSA_log.txt`

There are additional files in the /SSA directory that are flat files of the results reported in the log files:

RMSIPs

PAs

COs

Cosine Products

Vectorial Sum of Angles

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

v. JED Driver Input File Format: cPCA

```
-------------------------------------------------------------------------------------------
0.01
3.00
0     $multi
20    0     0     0
/working/directory/
Description           reference_PDB_file.pdb
residues.txt
original_PDB_Coordinates.txt        0
-------------------------------------------------------------------------------------------
```

Notes: This is a **whitespace** delineated file with **8 lines**.

Line 1 Field specifies the **percent** (double) of frames to remove from the data: 0.01 → 1%
Line 2 Field 1 specifies the **z-cutoff** (double): 3.00 → values with z_scores beyond |3.0|
Line 3 Field 1 specifies **read flag**, whether to **read PDB files** (0 or 1) → 0 = no
Field 2: specifies **multi flag**, if the PDB files are **Multi Chain** (0 or 1)
Line 4 Field 1 specifies the **number of cPCA modes** (integer) → 20 = top 20 modes
Field 2: specifies the **number of dPCA modes** (integer) →0 = none
Field 3: specifies the **number of dpPCA modes** (integer) →0 = none
Field 4: specifies the **number of Cartesian modes to Visualize**(integer) →0 = none
Line 5 Field 1 specifies the **working directory** (String)
Line 6 Field 1 specifies the job **description** (String)
Field 2 specifies the **reference PDB** (String)
Line 7 Field 1 specifies the **Cartesian residue list** (String)
Line 8 Field1 specifies the **coordinate matrix** (String)
Field 2 specifies the **reference column** (integer)

Key Points:

- The **Read** flag **MUST** be set to **0**
  - When the **Read** flag is set to **1**, all PCAs are turned off
- The **Multi** flag must be set to **0** for Single Chain PDBs with no Chain IDs
- The **Multi** flag must be set to **1** or "multi" for Multi Chain PDBs with Chain IDs
  - **Multi Chain PDBs must have unique chain identifiers for every chain**
  - **Missing chain identifiers will cause JED to crash**
- **The number of cPCA modes must NOT be 0.**
- **There must be a Cartesian residue list specified.**
- The number of dPCA modes and dpPCA modes must be zero.
- There must not be a Distance or Distance Pair residue list specified.
- If the subset you have chosen contains N residues, then you must NOT request more than 3N modes.
- If you request N cPCA modes then you can only visualize up to N top modes.

- **If number of cPCA modes > 0, then there must be a Cartesian residue list file!**

vi. Residue List Format for cPCA

The easiest way to create the residue list for cPCA is to edit the JED generated file that lists all residues found in the PDB files, in the proper format:

a. **Single Chain PDBs: "*All_PDB_Residues_JED.txt*"**

   ONE column of integers: residue numbers

b. **Multi Chin PDBs: "*All_PDB_Residues_Multi_JED.txt*"**

   TWO columns: Column 1 Strings, Column 2 integers (tab separated): Chain IDs, residue numbers

Note that all entries in the residue list are checked against the reference PDB file.

If a requested residue cannot be found in the reference file, then JED will crash with an error message stating that a requested residue could not be found.

D. Performing dPCA

i. Run command:

```
java -jar -d64 JED_Driver.jar "/path/JED_Driver.txt"
```

The working directory must contain: The coordinates matrix, the PDB reference file, and the residue list.
JED input file may be in the working directory.

The purpose of this type of run is to perform Essential Dynamics using dPCA based on **Q** and **R**. The user specifies the subset of interest for the analysis, which is typically less than 10 residues, by providing a residue list file. This task is simplified since JED has already created a list of all the residues in the protein. The user must simply edit this file. PCA is performed on the set of all to all distances calculated from the entered residues. The dPCA results are written to the sub-directory "dPCA". Note that for dPCA no transform is needed since internal distances are used for coordinates and no visualization can be done in JED for the distance modes. The directory dPCA has sub directories for the **Q** and **R** analysis as well as for the subspace analysis (SSA). Choosing more than ten residues for the dPCA analysis makes the interpretation of the results challenging as each component of the distance eigenvectors corresponds to an inter-residue distance pair. Often subsets with less than ten residues can be used to investigate experimental findings in critical areas like binding pockets or clefts.

ii. Root Output Files:

These are written to the **root** of the JED Results directory tree:

```
/working/directory/JED_Results_Description/
```

JED LOG providing a summary of the job parameters and results:

```
JED_Log.txt
```

iii. dPCA Output Files:

These are written to the **/dPCA subdirectory** of the JED Results directory tree:

```
/working/directory/JED_Results_Description/dPCA/
```

Subset PDB file:

```
ss_$num_res.PDB
```

coordinates Z-Score matrix:

```
ss_$num_res_distance_Z_scores.txt
```

list of coordinate variables adjusted, based on the **z-cutoff** parameter:

```
ss_$num_res_ outliers_per_variable.txt
```

distance residue stats:

```
ss_$num_res_distance_residue_stats.txt
```

distance Z-Score matrix:

```
ss_$num_res_distance_Z_scores.txt
```

all-to-all distances matrix:

```
ss_$num_res_all_to_all_distances.txt
```

distance variable outliers:

```
ss_$num_res_ outliers_per_variable.txt
```

centroids (means) of the variables:

`ss_$num_res_centroids_of_variables.txt`

standard deviations of the variables:

`ss_$num_res_std_devs_of_centered_variables.txt`

displacement vectors:

`ss_$num_res_delta_vectors.txt`

- **The Q output files are written to the /COV subdirectory of /dPCA** (Shown below)

`/working/directory/JED_Results_Description/dPCA/COV/`

- **The R output files are written to the /CORR subdirectory of /dPCA**

`/working/directory/JED_Results_Description/dPCA/CORR/`

covariance matrix:

`ss_$num_res_distance_COV_matrix.txt`

eigenvalues:

`ss_$num_res_distance_eigenvalues_COV.txt`

top eigenvalues:

`ss_$num_res_top_$num_of_dist_modes_distance_eigenvalues_COV.txt`

top eigenvectors:

`ss_$num_res_top_$num_of_dist_modes_distance_eigenvectors_COV.txt`

top pca modes and top weighted pca modes:

`ss_$num_res_top_$num_of_dist_modes_distance_pca_modes_COV.txt`
`ss_$num_res_top_$num_of_dist_modes_weighted_distance_pca_modes_COV.txt`

top square pca modes and top weighted square pca modes:

`ss_$num_res_top_$num_of_dist_modes_square_distance_pca_modes_COV.txt`
`ss_$num_res_top_$num_of_dist_modes_weighted_square_distance_pca_modes_COV.txt`

top DVPs and top weighted DVPs:

`ss_$num_res_top_$num_of_dist_modes_DVPs_COV.txt`
`ss_$num_res_top_$num_of_dist_modes_weighted_DVPs_COV.txt`

top normed DVPs and top weighted normed DVPs:

`ss_$num_res_top_$num_of_dist_modes_normed_DVPs_COV.txt`
`ss_$num_res_top_$num_of_dist_modes_weighted_normed_DVPs_COV.txt`

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

iv. SSA Output Files:

These are written to the **/SSA subdirectory** of /dPCA:
`/working/directory/JED_Results_Description/dPCA/SSA/`

The Fast SSA Iterated Log:

`JED_FSSA_Iterated_log.txt`

The SSA Log:

`JED_SSA_dim_$top_num_cart_modes_log.txt`

The Random SSA Log

`JED_Random_SSA_log.txt`

There are additional files in the /SSA directory that are flat files of the results reported in the log files:

RMSIPs

PAs

COs

Cosine Products

Vectorial Sum of Angles

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

v. JED Driver Input File Format: dPCA

```
-------------------------------------------------------------------------------------------
0.00
3.00
0     $multi
0     3     0     0
/working/directory/
Description           reference_PDB_file.pdb
residues_dist.txt
original_PDB_Coordinates.txt        0
-------------------------------------------------------------------------------------------
```

Notes: This is a **whitespace** delineated file with **8 lines**.

Line 1 Field 1 specifies the **percent** (double) of frames to remove from the data: 0.00 = none
Line 2 Field 1 specifies the **z-cutoff** (double): 3.00 → values with z_scores beyond |3.0|
Line 3 Field 1 specifies **read flag**, whether to **read PDB files** (0 or 1) → 0 = no
Field 2: specifies **multi flag**, if the PDB files are **Multi Chain** (0 or 1)
Line 4 Field 1 specifies the **number of cPCA modes** (integer) → 0 = none
Field 2: specifies the **number of dPCA modes** (integer) → 3 = top 3 modes
Field 3: specifies the **number of dpPCA modes** (integer) → 0 = none
Field 4: specifies the **number of Cartesian modes to Visualize** (integer) → 0 = none
Line 5 Field 1 specifies the **working directory** (String)
Line 6 Field 1 specifies the job **description** (String)
Field 2 specifies the **reference PDB** (String)
Line 7 Field 1specifies the **residue list** (String)
Line 8 Field 1 specifies the **coordinate matrix** (String)
Field 2 specifies the **reference column** (integer)

Key Points:

- The **Read** flag **MUST** be set to **0**
  - When the **Read** flag is set to **1**, all PCAs are turned off
- The **Multi** flag must be set to **0** for Single Chain PDBs with no Chain IDs
- The **Multi** flag must be set to **1** or "multi" for Multi Chain PDBs with Chain IDs
  - **Multi Chain PDBs must have unique chain identifiers for every chain**
  - **Missing chain identifiers will cause JED to crash**
- The number of cPCA modes must be 0.
- There must be no Cartesian residue list specified.
- **The number of dPCA modes must NOT be zero.**
- **There must be a Distance residue list specified.**
- The number of dpPCA modes must be zero.
- There must not be a Distance Pair residue list specified.
- If the subset you have chosen contains ***N*** residues, then you must NOT request more than ***N(N-1)/2*** modes.

- **If number of dPCA modes > 0, then there must be a Distance residue list file!**

vi. Residue List Format for dPCA

The easiest way to create the residue list for dPCA is to edit the JED generated file that lists all residues found in the PDB files, in the proper format:

c. Single Chain PDBs: "*All_PDB_Residues_JED.txt*"

ONE column of integers: residue numbers

d. Multi Chin PDBs: "*All_PDB_Residues_Multi_JED.txt*"

TWO columns: Column 1 Strings, Column 2 integers (tab separated): Chain IDs, residue numbers

Note that all entries in the residue list are checked against the reference PDB file.

If a requested residue cannot be found in the reference file, then JED will crash with an error message stating that a requested residue could not be found.

E. Performing dpPCA

i. Run command:

```
java -jar -d64 JED_Driver.jar "/path/JED_Driver.txt"
```

The working directory must contain: The coordinates matrix, the PDB reference file, and the residue list.
JED input file may be in the working directory.

The purpose of this type of run is to perform Essential Dynamics using dpPCA based on **Q** and **R**. The user specifies the set of **residue pairs** of interest for the analysis, by providing a residue pair list file. This file is a two column file for Single Chain PDBs in which the pairs of interest are listed. However, for Multi Chain PDBs, the file is four columns, the first two for the chain ID and residue number of residue one, and the third and fourth columns for the chain ID and residue number of the second residue. The dpPCA results are written to the sub-directory "dpPCA". Note that for dpPCA no transform is needed since internal distances are used for coordinates and no visualization can be done in JED for the distance modes. The residue pair method is much easier to interpret as the number of components in the eigenvectors is equal to the number of residue pairs specified. The directory dpPCA has sub directories for the **Q** and **R** analysis as well as for the subspace analysis (SSA). Very large subsets can be used to investigate experimental findings in critical areas like binding pockets or clefts. Unfortunately, like the dPCA results, the dpPCA results cannot be visualized as no simple mapping can be made to the residues.

ii. Root Output Files:

These are written to the **root** of the JED Results directory tree:
`/working/directory/JED_Results_Description/`

JED LOG providing a summary of the job parameters and results:

`JED_Log.txt`

iii. dpPCA Output Files:

These are written to the **/dpPCA subdirectory** of the JED Results directory tree:

`/working/directory/JED_Results_Description/dpPCA/`

coordinates Z-Score matrix:

`ss_$num_res_pairs_Distance_Pairs_distance_Z_scores.txt`

list of distance-pair variables adjusted, based on the **z-cutoff** parameter:

`ss_$num_res_pairs_Distance_Pairs_outliers_per_variable.txt`

distance residue stats from all the distance pairs in the **pair list**:

`ss_$num_res_pairs_Distance_Pairs_distance_residue_stats.txt`

distance Z-Score matrix from all the distance pairs in the **pair list**:

`ss_$num_res_pairs_Distance_Pairs_distance_Z_scores.txt`

distances matrix from all the distance pairs in the **pair list**:

`ss_$num_res_pairs_Distance_Pairs_distances.txt`

centroids (means) of the variables:

`ss_$num_res_pairs_Distance_Pairs_centroids_of_variables.txt`

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

standard deviations of the variables:

`ss_$num_res_pairs_Distance_Pairs_std_devs_of_centered_variables.txt`

displacement vectors:

`ss_$num_res_pairs_Distance_Pairs_delta_vectors.txt`

- **The Q output files are written to the /COV subdirectory of /dpPCA** (Shown below)

`/working/directory/JED_Results_Description/dpPCA/COV/`

- **The R output files are written to the /CORR subdirectory of /dpPCA**

`/working/directory/JED_Results_Description/dpPCA/CORR/`

covariance matrix:

`ss_$num_res_pairs_Distance_Pairs_distance_pair_COV_matrix.txt`

eigenvalues:

`ss_$num_res_pairs_Distance_Pairs_eigenvalues_COV.txt`

top eigenvalues:

`ss_$num_res_pairs_Distance_Pairs_top_$num_of_dist_modes_eigenvalues_COV.txt`

top eigenvectors:

`ss_$num_res_pairs_Distance_Pairs_top_$num_of_dist_modes_eigenvectors_COV.txt`

top pca modes and top weighted pca modes:

`ss_$num_res_pairs_Distance_Pairs_top_$num_of_dist_modes_pca_modes_COV.txt`
`ss_$num_res_pairs_Distance_Pairs_top_$num_of_dist_modes_weighted_pca_modes_COV.txt`

top square pca modes and top weighted square pca modes:

`ss_$num_res_pairs_Distance_Pairs_top_$num_of_dist_modes_square_pca_modes_COV.txt`
`ss_$num_res_pairs_Distance_Pairs_top_$num_of_dist_modes_weighted_square_pca_modes_COV.txt`

top DVPs and top weighted DVPs:

`ss_$num_res_pairs_Distance_Pairs_top_$num_of_dist_modes_DVPs_COV.txt`
`ss_$num_res_pairs_Distance_Pairs_top_$num_of_dist_modes_weighted_DVPs_COV.txt`

top normed DVPs and top weighted normed DVPs:

`ss_$num_res_pairs_Distance_Pairs_top_$num_of_dist_modes_normed_DVPs_COV.txt`
`ss_$num_res_pairs_Distance_Pairs_top_$num_of_dist_modes_weighted_normed_DVPs_COV.txt`

iv. SSA Output Files: Same as for the other types of PCA

These are written to the **/SSA subdirectory** of /dpPCA:

`/working/directory/JED_Results_Description/dpPCA/SSA/`

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

v. JED Driver Input File Format: dpPCA

```
-----------------------------------------------------------------------------------------
0.00
3.00
0     $multi
0     0     10     0
/working/directory/
Description           reference_PDB_file.pdb
residues_pairs.txt
original_PDB_Coordinates.txt        0
-----------------------------------------------------------------------------------------
```

Notes: This is a **whitespace** delineated file with **8 lines**.

Line 1 Field 1 specifies the **percent** (double) of frames to remove from the data: 0.00 = none
Line 2 Field 1 specifies the **z-cutoff** (double): 3.00 → values with z_scores beyond |3.0|
Line 3 Field 1 specifies **read flag**, whether to **read PDB files** (0 or 1) → 0 = no
Field 2: specifies **multi flag**, if the PDB files are **Multi Chain** (0 or 1)
Line 4 Field 1 specifies the **number of cPCA modes** (integer) → 0 = none
Field 2: specifies the **number of dPCA modes** (integer) → 0 = none
Field 3: specifies the **number of dpPCA modes** (integer) → 10 = top 10 modes
Field 4: specifies the **number of Cartesian modes to Visualize**(integer) →0 = none
Line 5 Field 1 specifies the **working directory** (String)
Line 6 Field 1 specifies the job **description** (String)
Field 2 specifies the **reference PDB** (String)
Line 7 Field 1specifies the **residue pair list** (String)
Line 8 Field 1 specifies the **coordinate matrix** (String)
Field 2 specifies the **reference column** (integer)

Key Points:

- The **Read** flag **MUST** be set to **0**
  - When the **Read** flag is set to **1**, all PCAs are turned off
- The **Multi** flag must be set to **0** for Single Chain PDBs with no Chain IDs
- The **Multi** flag must be set to **1** or "multi" for Multi Chain PDBs with Chain IDs
  - **Multi Chain PDBs must have unique chain identifiers for every chain**
  - **Missing chain identifiers will cause JED to crash**
- The number of cPCA modes must be 0.
- There must be no Cartesian residue list specified.
- The number of dPCA modes must be zero.
- There must not be a Distance residue list specified.
- **The number of dpPCA modes must NOT be zero.**
- **There MUST be a Distance Pair residue list specified.**
- If the subset you have chosen contains ***N*** pairs, then you must NOT request more than ***N*** modes.

- **If number of dpPCA modes > 0, then there must be a Distance Pair Residue list file!**

vi. Residue List Format for dpPCA

Residue pairs are specified one to a line in the residue pair list file:

a. **Single Chain PDBs: Two columns of integers, tab separated:**

   residue number1 residue number2

b. **Multi Chin PDBs: Four columns of strings and integers, tab separated:**

   ChainID1 Res_Number1 Chain ID2 Res_Number2

Note that all entries in the residue list are checked against the reference PDB file.

If a requested residue cannot be found in the reference file, then JED will crash with an error message stating that a requested residue could not be found.

F. Debugging Crashes Part II:

Things that will generally make your life miserable…

i. Simple mistakes:

Did you set the Read and Multi flags correctly?
Did you request cPCA but not specify a Cartesian residue list?
Did you request dPCA but not specify a Distance residue list?
Did you request dpPCA but not specify a Distance Residue Pair List?
Did you set the number of modes appropriately?
Are you requesting to read PDBs when you should be specifying a coordinate matrix?
Are you requesting residues that are not in the reference PDB?

ii. Subtle mistakes:

Did you request more modes than actually exist?

For example, for cPCA

if your Cartesian subset contains 12 residues and you ask for 50 modes, then you are going to crash JED:
Because there are only 36 Cartesian modes in total.

For example, for dPCA

if your Distance subset contains 5 residues and you request 15 modes, then you are going to crash JED:
Because there are only 10 distance modes in total.

For example, for dpPCA

if your Distance Pair List contains 5 pairs and you request 10 modes, then you are going to crash JED:
Because there are only 5 distance-pair modes in total.

If your trajectory has not equilibrated, then you must address the problem of outliers. If you do not, then the Q and R matrices will be highly ill-conditioned and may cause the eigenvalue decomposition to fail. You can check the variables in statistics packages that compute the Kaiser-Myer-Olkin (KMO) statistic as well as the Measure of Sampling Adequacy (MSA) for each coordinate variable to critically assess your data. If it is not well suited for PCA, you can condition the variables by setting the z-cutoff in JED between 2.0 and 3.0 when running your jobs. This type of conditioning is by far not very sophisticated, but it has the effect of lowering the condition numbers of Q and R as well as un-dilating the high and low regions of the eigenspectrum. In particular, it does not alter the ordinality of the eigenvalues, but does correct the distortion that arises from under sampling when trying to estimate the population covariance matrix from the sample covariance matrix.

G. Performing Cartesian Mode Visualization

To activate cPCA mode visualization, you must be running a job with cPCA selected. To generate the output files, you only need to **set the number of modes VIZ ≤ number of cPCA modes**, and **set the mode amplitude > 0**.

Note: Setting the modes VIZ flag to zero turns off the visualization feature.

******* If you set the value of modes VIZ > 0, then you MUST provide the mode amplitude! *******

The outputs to this type of job include all the structures for the top modes chosen for visualization. JED will permute the reference structure for a given subset along the top eigenvectors selected for visualization and output structures (PDBs) that capture one cycle of this motion. The amplitude of the motion is determined by the value of the **$mode_amplitude**, whose default value is 1.0, and can be adjusted as necessary. Setting the value too high will cause Visualization software like Pymol to break the ribbon diagrams of the structures. Ultimately, this is controlled by the magnitude of the eigenvector components for any given residue. Setting the mode amplitude between 1 and 3 is usually safe, but for proteins with highly mobile regions like loops, you may need to choose a smaller mode amplitude. This is done for both the top **Q** and **R** modes. Additionally, Pymol scripts are generated to animate those structures into a movie for better analysis of the physical meanings of the top modes.

These files will be located in the /VIZ subdirectory of the root of the JED results tree:
`/working/directory/JED_Results_Description/VIZ/`

The **Q** results will be in the subdirectory /COV and the **R** results will be in the subdirectory /CORR.

H. Performing Multiple PCAs

The working directory must contain: The coordinates matrix, the PDB reference file, and the residue lists.
It may also contain the JED input file

JED is capable of doing cPCA (with or without visualization), dPCA, and dPCA simultaneously.
All outputs are delivered as discussed for the individual components.

JED expects the input file to follow the following format regarding the order of the residue lists:

if cPCA then Cartesian Residue List
if dPCA then Distance Residue List
if dpPCA then Distance Pair Residue List

An important advantage of JED is that it is highly configurable to perform many types of Essential Dynamics analysis concurrently. Combined with cluster resources or just using the batch feature (discussed in the next section) allows a user to process a great deal of data efficiently.

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

i. JED Driver Input File Format: cPCA, dPCA

```
-----------------------------------------------------------------------------------------
0.00
3.00
0     $multi
20    3     0     0
/working/directory/
Description           reference_PDB_file.pdb
residues_cartesian.txt
residues_dist.txt
original_PDB_Coordinates.txt        0
-----------------------------------------------------------------------------------------
```

Key Points:

- The **Read** flag **MUST** be set to **0**
- The **Multi** flag must be set to **0** for Single Chain PDBs with no Chain IDs
- The **Multi** flag must be set to **1** or "multi" for Multi Chain PDBs with Chain IDs
- If the number of cPCA modes > 0, then there must be a Cartesian residue list specified
- If the number of dPCA modes > 0, then there must be a Distance residue list specified
- If the number of dpPCA modes > 0, then there must be a Distance Pair residue list specified

ii. JED Driver Input File Format: cPCA, dPCA, dpPCA, VIZ

```
-----------------------------------------------------------------------------------------
0.00
3.00
0     $multi
20    5     3     2     1.0
/working/directory/
Description           reference_PDB_file.pdb
residues_cartesian.txt
residues_dist.txt
residue_pairs.txt
original_PDB_Coordinates.txt        0
-----------------------------------------------------------------------------------------
```

Key Points:

- The **Read** flag **MUST** be set to **0**
- The **Multi** flag must be set to **0** for Single Chain PDBs with no Chain IDs
- The **Multi** flag must be set to **1** or "multi" for Multi Chain PDBs with Chain IDs
- If the number of cPCA modes > 0, then there must be a Cartesian residue list specified
- If the number of dPCA modes > 0, then there must be a Distance residue list specified
- If the number of dpPCA modes > 0, then there must be a Distance Pair residue list specified

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

V. USING JED BATCH DRIVER

The batch version is identical to the non-batch version with the exception of the format of the input file.

A. JED Batch Driver Input File Format: The Preliminary Run

```
-------------------------------------------------------------------------------------------------------
$num_of_jobs
0.00
0.00
1     $multi
0     0     0     0
**************************************************************************
/working/directory/job1/
Description1                reference_PDB_file1.pdb
**************************************************************************
/working/directory/job2/
Description2                reference_PDB_file2.pdb
**************************************************************************
-------------------------------------------------------------------------------------------------------
```

Notes: This is a whitespace delineated file.

Line 1 Field 1 specifies the **$num_of_jobs** (integer) for the batch.
Line 2 Field 1 specifies the **percent** (double): 0.00 → NONE
Line 3 Field 1 specifies the **z-cutoff** (double): 0.00 → NO ADJUSTMENT
Line 4 Field 1 specifies the **read flag**, whether to **read PDB files** (0 or 1) → 1 = yes
Field 2 specifies the **multi flag**, if the PDB files are **Multi** Chain (0 or 1) → 0 = no
Line 5 Field 1 specifies the **number of cPCA** modes (integer) → 0 = none
Field 2 specifies the **number of dPCA** modes (integer) → 0 = none
Field 3 specifies the **number of dpPCA** modes (integer) → 0 = none
Field 4 specifies the **number of Cartesian modes to Visualize** (integer) → 0 = none
Line 6 is a **separator line** between the batch parameters and the job parameters **********
Line 7 Field 1 specifies the **working directory** (String) for job one
Line 8 Field 1 specifies the job **description** (String) for job one
Field 2 specifies the **reference PDB** (String) for job one
Line 9 is a **separator line** between sets of job parameters **********
Line 10 Field 1 specifies the **working directory** (String) for job one
Line 11 Field 1 specifies the job **description** (String) for job two
Field 2 specifies the **reference PDB** (String) for job two
Line 12 is a **separator line** between sets of job parameters **********

Key Points:

- **Be sure that the number of jobs matches the number of job inputs.**
- The **Read** flag **MUST** be set to **1** or **"read"**
- The **Multi** flag must be set to **0** for Single Chain PDBs with no Chain IDs
- The **Multi** flag must be set to **1** or "multi" for Multi Chain PDBs with Chain IDs
- **Make sure to use separator lines after the batch parameters, between jobs, and after the last job.**

B. JED Batch Driver Input File Format: cPCA

```
-------------------------------------------------------------------------------------------
$num_of_jobs
0.01
3.00
0     0
20    0     0     0
*****************************************************************************
/working/directory1/
Description1          reference_PDB_file1.pdb
residues1.txt
original_PDB_Coordinates.txt        0
*****************************************************************************
/working/directory2/
Description2          reference_PDB_file2.pdb
residues2.txt
original_PDB_Coordinates.txt        0
*****************************************************************************
-------------------------------------------------------------------------------------------
```

Notes:

Line 1 Field 1 specifies the **$num_of_jobs** (integer) for the batch
Line 2 Field 1 specifies the **percent** (double) of frames to remove from the data: 0.01 → 1%
Line 3 Field 1 specifies the **z-cutoff** (double) for adjusting outliers: 3.00 → values beyond |3.0|
Line 4 Field 1 specifies the **read flag**, whether to **read PDB files** (0 or 1) → 0 = no
Field 2 specifies the **multi flag**, if the PDB files are **Multi** Chain (0 or 1) → 0 = no
Line 5 Field 1 specifies the **number of cPCA** modes (integer) → 20 = top 20 modes
Field 2 specifies the **number of dPCA** modes (integer) → 0 = none
Field 3 specifies the **number of dpPCA** modes (integer) → 0 = none
Field 4 specifies the **number of Cartesian modes to Visualize**(integer) → 0 = none
Line 6 is a **separator line** between the batch parameters and the job parameters **********
Line 7 Field 1 specifies the **working directory** (String) for job one
Line 8 Field 1 specifies the job **description** (String) for job one
Field 2 specifies the **reference PDB** (String) for job one
Line 9 Field 1 specifies the **residue list** (String) for job one
Line 10 Field 1 specifies the **coordinate matrix** (String) for job one
Field 2 specifies the **reference column** (integer) for job one
Line 11 is a **separator line** between sets of job parameters **********
Line 12 Field 1 specifies the **working directory** (String) for job two
Line 13 Field 1 specifies the job **description** (String) for job two
Field 2 specifies the **reference PDB** (String) for job two
Line 14 Field 1 specifies the **residue list** (String) for job two
Line 15 Field 1 specifies the **coordinate matrix** (String) for job two
Field 2 specifies the **reference column** (integer) for job two
Line 16 is a **separator line** between sets of job parameters **********

C. JED Batch Driver Input File Format: dPCA

```
-------------------------------------------------------------------------------------------
$num_of_jobs
0.01
3.00
0     0
0     3     0     0
*****************************************************************************
/working/directory1/
Description1          reference_PDB_file1.pdb
residues_dist1.txt
original_PDB_Coordinates1.txt       0
*****************************************************************************
/working/directory2/
Description2          reference_PDB_file2.pdb
residues_dist2.txt
original_PDB_Coordinates2.txt       0
*****************************************************************************
-------------------------------------------------------------------------------------------
```

Notes:

Line 1 Field 1 specifies the **$num_of_jobs** (integer) for the batch
Line 2 Field 1 specifies the **percent** (double) of frames to remove from the data: 0.01 → 1%
Line 3 Field 1 specifies the **z-cutoff** (double) for adjusting outliers: 3.00 → values beyond |3.0|
Line 4 Field 1 specifies the **read flag**, whether to **read PDB files** (0 or 1) → 0 = no
Field 2 specifies the **multi flag**, if the PDB files are **Multi** Chain (0 or 1) → 0 = no
Line 5 Field 1 specifies the **number of cPCA** modes (integer) → 0 = none
Field 2 specifies the **number of dPCA** modes (integer) → 3 = top 3 modes
Field 3 specifies the **number of dpPCA** modes (integer) → 0 = none
Field 4 specifies the **number of Cartesian modes to Visualize**(integer) → 0 = none
Line 6 is a **separator line** between the batch parameters and the job parameters **********
Line 7 Field 1 specifies the **working directory** (String) for job one
Line 8 Field 1 specifies the job **description** (String) for job one
Field 2 specifies the **reference PDB** (String) for job one
Line 9 Field 1 specifies the **distance residue list** (String) for job one
Line 10 Field 1 specifies the **coordinate matrix** (String) for job one
Field 2 specifies the **reference column** (integer) for job one
Line 11 is a **separator line** between sets of job parameters **********
Line 12 Field 1 specifies the **working directory** (String) for job two
Line 13 Field 1 specifies the job **description** (String) for job two
Field 2 specifies the **reference PDB** (String) for job two
Line 14 Field 1 specifies the **distance residue list** (String) for job two
Line 15 Field 1 specifies the **coordinate matrix** (String) for job two
Field 2 specifies the **reference column** (integer) for job two
Line 16 is a **separator line** between sets of job parameters **********

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

D. JED Batch Driver Input File Format: dpPCA

```
-------------------------------------------------------------------------------------------
$num_of_jobs
0.01
3.00
0     0
0     0     3     0
*****************************************************************************
/working/directory1/
Description1          reference_PDB_file1.pdb
residues_pairs1.txt
original_PDB_Coordinates1.txt       0
*****************************************************************************
/working/directory2/
Description2          reference_PDB_file2.pdb
residues_pairs2.txt
original_PDB_Coordinates2.txt       0
*****************************************************************************
-------------------------------------------------------------------------------------------
```

Notes:

Line 1 Field 1 specifies the **$num_of_jobs** (integer) for the batch
Line 2 Field 1 specifies the **percent** (double) of frames to remove from the data: 0.01 → 1%
Line 3 Field 1 specifies the **z-cutoff** (double) for adjusting outliers: 3.00 → values beyond |3.0|
Line 4 Field 1 specifies the **read flag**, whether to **read PDB files** (0 or 1) → 0 = no
Field 2 specifies the **multi flag**, if the PDB files are **Multi** Chain (0 or 1) → 0 = no
Line 5 Field 1 specifies the **number of cPCA** modes (integer) → 0 = none
Field 2 specifies the **number of dPCA** modes (integer) → 3 = none
Field 3 specifies the **number of dpPCA** modes (integer) → 0 = top 3 modes
Field 4 specifies the **number of Cartesian modes to Visualize**(integer) → 0 = none
Line 6 is a **separator line** between the batch parameters and the job parameters **********
Line 7 Field 1 specifies the **working directory** (String) for job one
Line 8 Field 1 specifies the job **description** (String) for job one
Field 2 specifies the **reference PDB** (String) for job one
Line 9 Field 1 specifies the **residue pair list** (String) for job one
Line 10 Field 1 specifies the **coordinate matrix** (String) for job one
Field 2 specifies the **reference column** (integer) for job one
Line 11 is a **separator line** between sets of job parameters **********
Line 12 Field 1 specifies the **working directory** (String) for job two
Line 13 Field 1 specifies the job **description** (String) for job two
Field 2 specifies the **reference PDB** (String) for job two
Line 14 Field 1 specifies the **residue pair list** (String) for job two
Line 15 Field 1 specifies the **coordinate matrix** (String) for job two
Field 2 specifies the **reference column** (integer) for job two
Line 16 is a **separator line** between sets of job parameters **********

E. JED Batch Driver Input File Format: cPCA, dPCA, dpPCA, VIZ

```
-------------------------------------------------------------------------------------------
$num_of_jobs
0.01
3.00
0     $multi
20    5     3     2     1.0
*******************************************************************************
/working/directory1/
Description1          reference_PDB_file1.pdb
residues1.txt
residues_dist1.txt
residues_pairs1.txt
original_PDB_Coordinates.txt        $ref_col
*******************************************************************************
/working/directory2/
Description2          reference_PDB_file2.pdb
residues2.txt
residues_dist2.txt
residues_pairs2.txt
original_PDB_Coordinates.txt        $ref_col
*******************************************************************************
-------------------------------------------------------------------------------------------
```

Key Points:

- **Be sure that the number of jobs matches the number of job inputs**
- The **Read** flag **MUST** be set to **0**
- The **Multi** flag must be set to **0** for Single Chain PDBs with no Chain IDs
- The **Multi** flag must be set to **1** or "multi" for Multi Chain PDBs with Chain IDs
- Make sure to use **separator lines** after the batch parameters, between jobs, and after the last job
- Make sure to specify the residue lists in the **correct order**: **cPCA, dPCA, dpPCA**
- Make sure to specify the **$mode_amplitude** if you request cPCA mode visualization

F. Debugging Batch Crashes:

Handling batch problems is more difficult than for single jobs. Any problem in any job can cause a crash.
Thus, it is good to track the standard out and errors streams to record the cause of any problems.
Also, you will know which jobs ran successfully so that you can edit the batch input file to finish the undone jobs.

iii. Simple mistakes:

Are the **Read** and **Multi** flags set correctly?
Is the **number format** correct? (20.0 will NOT parse as an integer)
Do ALL the working directories, residue lists, and PDB reference files **exist**?
Do ALL the working directories end in "**/**" or "****"?
Does the working directory contain **ALL of the required files**? (3 PCAs = 3 residue list input files)

iv. More subtle mistakes:

Do any of the jobs have residue sets that will not allow the batch parameters to apply?
For example, Too few residues for the number of modes specified.
Does the directory contain PDB files in **non-standard format** or with fractional **occupancy** data?
Does the reference PDB file correspond to the trajectory? (JED must map the ref pdb to the coords matrix)
The directory contains Multi Chain PDB files with missing chain IDs.

VI. Additional Types of Analysis

A. Pooling Data:

It is often useful to pool trajectory statistics. This can be done in JED by combining coordinate files and then performing the usual analysis. To combine the coordinate files, there is a utility program called **JED_Pool_Data.java** that will combine multiple matrices into one. Each matrix is appended to the last column of the preceding matrix. Of courses, the number of rows in the coordinate files must match. The matrices to combine are specified by an input file called **pool.txt** that the user must construct correctly.

i. Run Command:

```
java -jar -d64 JED_Pool_Data.jar "/path/to/pool.txt"
```

ii. Input File format:

LINE 1 specifies the number of jobs (integer)
LINE 2 is a separator line ********
Then for each job you must specify the following:**
The number of matrices to combine (integer)
The output directory (string ending in "/" or "\\")
The path to each matrix (String)
A separator line ********

Sample "pool.txt":

```
-------------------------------------------------------------------------------------------
2
**********************************************************************************
3
/output/directory1/
/path/to/first/coords/matrix/original_PDB_Coordinates.txt
/path/to/second/coords/matrix/original_PDB_Coordinates.txt
/path/to/third/coords/matrix/original_PDB_Coordinates.txt
**********************************************************************************
3
/output/directory2/
/path/to/first/coords/matrix/original_PDB_Coordinates.txt
/path/to/second/coords/matrix/original_PDB_Coordinates.txt
/path/to/third/coords/matrix/original_PDB_Coordinates.txt
**********************************************************************************
-------------------------------------------------------------------------------------------
```

Notes:
This file specifies 2 jobs, with 3 matrices to combine for each job.
Be sure that each path and matrix file exists.
---

iii. Output File format:

The output is a single, augmented matrix with the same number of rows as the composite matrices and columns equal to the sum of all columns in the composite matrices.

The output file name is: **pooled_matrix_*$number_of_input_matrices*.txt** (`pooled_matrix_3.txt` for the example)**.**

B. Subspace Analysis:

Once JED Driver has been run on multiple trajectories as well as pooled trajectories, an analysis can be done to compare how similar the essential subspaces derived from those trajectories are to each other. JED contains a program called **Subspace_Analysis.java** along with 3 driver programs that perform those functions. The core program takes as input two matrices of eigenvectors derived from PCA (or NMA, ANM, etc.). The matrices must have the same number of rows and columns, meaning the vectors being compared come from the same vector space and that the subspaces have the same dimensions. For example, in an analysis of lysozyme you might choose to process 20 cPCA modes while examining 10 different experimental conditions plus pooled data. As long as all the subsets in the analysis are the same then all the 20 dimensional subspaces can be compared.

Like most of the JED programs, the subspace analysis program driver reads an input file called **SSA.txt** to obtain runtime information. This file must be constructed properly to perform the analysis correctly. The three driver programs are **SSA_Driver.java**, **FSSA_Driver.java**, and **FSSA_Iterated_Driver.java** and are different in how much analysis is requested. The SSA_Driver gives full outputs for non-iterated subspace comparison including both log files and individual flat files. The FSSA_Driver is a light-weight version with only RMSIP and PA output in the log files. The Iterated version performs a recursive variation of the above where all equidimensional subspaces are compared up to the size that was provided, for example, from 1 to 20 by step-size 1 for a 20 column input file.

i. Run Commands:

```
java -jar -d64 SSA_Driver.jar “/path/to/SSA.txt”
java -jar -d64 FSSA_Driver.jar “/path/to/SSA.txt”
java -jar -d64 FSSA_Iterated_Driver.jar “/path/to/SSA.txt”
```

ii. Input File format:

ALL three drivers use the same input file, only the outputs are different.

The format for the file shown below is:

LINE 1: Number_of_Jobs (integer)
LINE 2: Output_Directory (string ending in "/" or "\\")
LINE 3: Batch_Decription (string)
LINE 3: Separator Line ********************************
THEN FOR EACH JOB:
Description (string)
Directory1 (string ending in "/" or "\\") Name1 (string)
Directory2 (string ending in "/" or "\\") Name2 (string)
**Separator Line **

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

Sample "SSA.txt":

```
--------------------------------------------------------------------------------------------
5
/output/directory1/
MV_PCA_Model_Test
********************************************************************************************
MV1
/path/to/first/eigenvector/matrix/                ss_946_top_20_eigenvectors_COV.txt
/path/to/second/eigenvector/matrix/               ss_946_top_20_eigenvectors_CORR.txt
********************************************************************************************
MV2
/path/to/first/eigenvector/matrix/                ss_946_top_20_eigenvectors_COV.txt
/path/to/second/eigenvector/matrix/               ss_946_top_20_eigenvectors_CORR.txt
********************************************************************************************
MV3
/path/to/first/eigenvector/matrix/                ss_946_top_20_eigenvectors_COV.txt
/path/to/second/eigenvector/matrix/               ss_946_top_20_eigenvectors_CORR.txt
********************************************************************************************
MV4
/path/to/first/eigenvector/matrix/                ss_946_top_20_eigenvectors_COV.txt
/path/to/second/eigenvector/matrix/               ss_946_top_20_eigenvectors_CORR.txt
********************************************************************************************
MV5
/path/to/first/eigenvector/matrix/                ss_946_top_20_eigenvectors_COV.txt
/path/to/second/eigenvector/matrix/               ss_946_top_20_eigenvectors_CORR.txt
********************************************************************************************
--------------------------------------------------------------------------------------------
```

Supplemental Material: JED: Java Essential Dynamics, Charles David and Donald Jacobs

APPENDIX

I. Input File Formats

A. JED_Driver.txt

```
-------------------------------------------------------------------------------------------
$percent
$z-cut
$read          $multi
$c_modes       $d_modes     $dp_modes     $viz_modes     $mode_amplitude
$working_directory
$description   $reference_PDB
$residues_cartesian                        (optional)
$residues_dist.txt                         (optional)
$residue_pairs.txt                         (optional)
$PDB_Coordinates      $ref_col             (optional)
-------------------------------------------------------------------------------------------
```

B. JED_Batch_Driver.txt

```
-------------------------------------------------------------------------------------------
$num_jobs
$percent
$z-cut
$read          $multi
$c_modes       $d_modes     $dp_modes     $viz_modes     $mode_amplitude
*************************************************************************
$working_directory
$description   $reference_PDB
$residues_cartesian                        (optional)
$residues_dist.txt                         (optional)
$residue_pairs.txt                         (optional)
$PDB_Coordinates      $ref_col             (optional)
*************************************************************************
-------------------------------------------------------------------------------------------
```

C. Residue List file for cPCA and dPCA: Single Chain PDBs

```
-------------------------------------------------------------------------------------------
1
2
3
7
8
9
10
23
24
25
-------------------------------------------------------------------------------------------
```

D. Residue List file for cPCA and dPCA: Multi Chain PDBs

```
A       1
A       2
A       3
A       7
A       8
A       9
A       10
B       1
B       2
B       3
B       4
B       5
```

E. Residue Pair List file for dpPCA: Single Chain PDBs

```
2               27
5               32
12              57
18              80
36              101
```

F. Residue Pair List file for dpPCA: Multi Chain PDBs

```
A               2               A               27
A               5               A               32
A               12              B               57
B               18              B               80
B               36              B               101
```

G. pool.txt

```
$num_jobs --> (repeat job declaration $num_jobs times)
*******************************************************************************
$num_of_matrices_to_combine
$output_directory
$path_to_coords_matrix --> (repeat line $num_of_matrices_to_combine times)
*******************************************************************************
```

H. SSA.txt

```
$num_jobs --> (repeat job declaration $num_jobs times)
$output_directory
$batch_description
*******************************************************************************
$job_description
$path_to_first_eigenvector_file
$path_to_second_eigenvector_file
*******************************************************************************
```

II. Output File Formats

A. Sample JED Log file: cPCA, dPCA, dpPCA, VIZ

```
-----------------------------------------------------------------------------------------
JED: Java Essential Dynamics version 1.0
Job Description: TEST
Working directory: /JED_1.0/JED_Test/
Output directory: /JED_1.0/JED_Test/JED_RESULTS_TEST/

The alpha carbon coordinates were obtained from coordinates matrix file: original_PDB_Coordinates.txt
The dimension of the coordinates matrix is = 2838 by 101
Total number of residues in matrix = 946
Total number of conformations in matrix = 101
Transformed PDB coordinates obtained by quaternion least-squares alignment to the reference structure.
PDB reference structure is: MVb_A_B_ATP.pdb
Reference Column in matrix = 0

Performing cPCA, top 5 modes.

Residue list for Cartesian subset:  residues.txt
Number of residues in Cartesian subset: 33
The transformed data was trimmed by removing 1.0 percent of the samples based on conformation RMSD.
The coordinate values with Z-scores beyond |2.0| were set to their mean value.
Condition Number (CN_Q) of the covariance matrix (Q) = 180,035,158,412
LOG of CN_Q = 11
Trace of Q = 2
Condition Number (CN_R) of the correlation matrix (R) = 23,842,326,228
LOG of CN_R = 10
Trace of R = 99

The PDB file with B-factors replaced by residue RMSDs: ss_33_RMSF_edited.pdb

Performing dPCA, top 3 modes.

Residue list for the All-to-All Residue Distances subset:  residues_dist.txt
Number of residues in the All-to-All Residue Distances subset: 5
The distance values with Z-scores beyond |2.0| were set to their mean value.
Condition Number (CN_Q) of the All-to-All Distance Covariance matrix (Q_dist) = 844,983
LOG of CN_Q = 6
Trace of Q_dist = 2
Condition Number (CN_R) of the All-to-All Distance Correlation matrix (R_dist) = 18,546
LOG of CN_R = 4
Trace of R_dist = 10

MEANs and STANDARD DEVIATIONs for the All-to-All Residue Distances:

Res1        Res2            Mean            Std_Dev
A1          A2              3.886           0.002
A1          A3              6.510           0.643
A1          B9              55.858          0.755
A1          B10             58.193          0.738
A2          A3              3.877           0.002
A2          B9              53.370          0.487
A2          B10             55.693          0.510
A3          B9              50.474          0.299
A3          B10             52.940          0.288
B9          B10             3.857           0.002

Performing dpPCA, top 4 modes.

Residue Pair list:  residue_pairs.txt
Number of residues pairs: 5
The distance values with Z-scores beyond |2.0| were set to their mean value.
Condition Number (CN_Q) of the Residue Pair Distance Covariance matrix (Q_rpd) = 5,405,113
LOG of CN_Q = 7
Trace of Q_rpd = 1
Condition Number (CN_R) of the Residue Pair Distance Correlation matrix (R_rpd) = 7
LOG of CN_R = 1
Trace of R_rpd = 5
```

```
MEANs and STANDARD DEVIATIONs for the Residue Pair Distances:

Res1        Res2            Mean            Std_Dev
A1          B6              51.388          0.783
A2          B7              48.059          0.477
A3          B8              48.222          0.286
A4          B9              47.437          0.003
A5          B10             47.021          0.000

Performing Cartesian Mode Visualization on top  1  cPCA modes.

Sets of 20 structures were generated to animate each cPCA mode, using both Q and R PCA models.
MODE AMPLITUDE: 1.000

Analysis completed: 2013-12-29 03:36:28
--------------------------------------------------------------------------------
```

B. Sample PDB READ Log file:

```
1A6N.pdb
1A6N_froda_00000001.pdb
1A6N_froda_00000002.pdb
1A6N_froda_00000003.pdb
1A6N_froda_00000004.pdb
1A6N_froda_00000005.pdb
1A6N_froda_00000006.pdb
1A6N_froda_00000007.pdb
1A6N_froda_00000008.pdb
1A6N_froda_00000009.pdb
1A6N_froda_00000010.pdb
1A6N_froda_00000011.pdb
1A6N_froda_00000012.pdb
1A6N_froda_00000013.pdb
1A6N_froda_00000014.pdb
1A6N_froda_00000015.pdb
1A6N_froda_00000016.pdb
1A6N_froda_00000017.pdb
1A6N_froda_00000018.pdb
1A6N_froda_00000019.pdb
1A6N_froda_00000020.pdb
1A6N_froda_00000021.pdb
1A6N_froda_00000022.pdb
1A6N_froda_00000023.pdb
1A6N_froda_00000024.pdb
1A6N_froda_00000025.pdb
```

C. Sample SSA Log File:

```
-----------------------------------------------------------------------------------------
Top_COV_Eigenvectors
Rows: 99
Cols: 5
Top_CORR_Eigenvectors
Rows: 99
Cols: 5

Projections file written to:
/workspace/jed/JED_RESULTS_TEST/cPCA/SSA/TEST_projections_dim_5.txt

Cumulative overlaps 1 --> 2 file written to:
/workspace/jed/JED_RESULTS_TEST/cPCA/SSA/TEST_CO_1_2_dim_5.txt

Cumulative overlaps 2 --> 1 file written to:
/workspace/jed/JED_RESULTS_TEST/cPCA/SSA/TEST_CO_2_1_dim_5.txt

Principle Angles file written to:
/workspace/jed/JED_RESULTS_TEST/cPCA/SSA/TEST_PA_dim_5.txt

Cosine Products file written to:
/workspace/jed/JED_RESULTS_TEST/cPCA/SSA/TEST_Cosine_Products_dim_5.txt

Vectorial sums of angles file written to:
/workspace/jed/JED_RESULTS_TEST/cPCA/SSA/TEST_Vector_sums_of_Angles_dim_5.txt

The absolute projections of each vector in subspace 1 with each vector in subspace 2 are:

    0.51    0.07    0.14    0.03    0.05
    0.03    0.59    0.00    0.03    0.04
    0.11    0.10    0.29    0.07    0.30
    0.08    0.01    0.10    0.38    0.07
    0.03    0.11    0.24    0.20    0.05

The cumulative overlaps CO_5 for each vector in subspace 1 with all the vectors in subspace 2 are:
Vector 1 0.540
Vector 2 0.593
Vector 3 0.451
Vector 4 0.407
Vector 5 0.339

The cumulative overlaps CO_5 for each vector in subspace 2 with all the vectors in subspace 1 are:
Vector 1 0.533
Vector 2 0.614
Vector 3 0.417
Vector 4 0.436
Vector 5 0.321

The RMSIP score is 0.475

The principle angles (in degrees) are:
PA1             37
PA2             41
PA3             48
PA4             52
PA5             61

The cosine products (in degrees) are:
CP1             37
CP2             55
CP3             73
CP4             82
CP5             88

The vectorial sums of angles (in degrees) are:
VS1             37
VS2             55
VS3             73
VS4             90
VS5             109

Maximum possible angle between two subspaces of this dimension is 201 degrees

Analysis completed: 2013-12-29 03:36:27
-----------------------------------------------------------------------------------------
```

D. Sample FSSA Iterated Log File:

```
-------------------------------------------------------------------------------------------
Principle Angle Spectra file written to:
/workspace/jed/JED_RESULTS_TEST/cPCA/SSA/TEST_PA_Spectra.txt

RMSIPs file written to:
/workspace/jed/JED_RESULTS_TEST/cPCA/SSA/TEST_RMSIPs.txt


RMSIPs:
Dim 1          0.513
Dim 2          0.555
Dim 3          0.498
Dim 4          0.478
Dim 5          0.475

The PA spectra for the range of subspaces are:

   44    0    0    0    0
   39   44    0    0    0
   37   42   61    0    0
   37   42   49   64    0
   37   41   48   52   61

Analysis completed: 2013-12-29 03:36:27
-------------------------------------------------------------------------------------------
```