JED: A Java Essential Dynamics Program for

Comparative Analysis of Protein Trajectories

Charles C. David

University of North Carolina at Charlotte

Department of Bioinformatics and Computational Biology

Donald J. Jacobs

University of North Carolina at Charlotte

Department of Physics and Optical Science

NOTE: April 1 is the deadline. I would like to see a first draft much sooner than near the deadline. Something like March 8 would be good. The paper length needs to be 4 pages, which is less than 30 hours of work for a 1st draft. Note that originally we were thinking this would be a 3-page paper. Should be possible.

Background

Essential Dynamics (ED) is a common application of principal component analysis (PCA) to extract biologically relevant motions from atomic trajectories of proteins. A covariance (Q) or correlation (R) based form of PCA for a selected set of atoms is applied to yield PCA modes (eigenvectors) and their eigenvalues. The displacement vectors (DV) relative to a reference structure are projected onto the top set of modes to produce a set of principal components, which are used to cluster conformations. The subspace defined by a relatively small set of PCA modes with largest eigenvalues quantifies conformational space explored by a protein. In bioinformatics studies, it is necessary to make subspace comparisons over a set of different protein trajectories for similarity assessment, but software to facilitate this comparative-analysis is needed.

Results

We developed the Java Essential Dynamics (JED) package to perform complete ED for multiple protein trajectories. JED reads the trajectory data from sets of PDB files or from a matrix of atomic coordinates. The PDB files may be single or multi-chained. A set of residues can be chosen that need not be contiguous. The analysis is based on alpha carbon atoms from each of the selected residues. Cartesian-based coordinates (cPCA) or internal distance coordinates (dPCA) can be applied with Q and R based PCA. JED outputs results as text files for further processing using generic graphing software. Key data include the transformed coordinates, conformation and residue RMSDs, PCA modes with their eigenvalues and DV projections onto the top principal components. A set of PDB files is generated along with a Pymol™ script to visualize each of the top modes derived from Q and R cPCA as either a static picture or as a movie. For comparative studies, JED compares subspaces defined by the top eigenvectors using several metrics to quantify the similarity/overlap of high dimensional vector spaces.

Conclusions

We present JED, a new Java package that encourages best-practices ED, allowing one to perform comparative studies over multiple protein trajectories to extract essential motions for the entire protein, user-defined sub-regions of interest, and to interrogate the significance of the results.

**Keywords**

Essential Dynamics, PCA, distance PCA, vector space, subspace, RMSIP, principal angle

**Background**

Protein dynamics is the change in molecular structure, or conformation, as a function of time. Accessible molecular motions over a wide range of time and spatial scales can be represented by a vector space that spans a number of dimensions equal to the number of degrees of freedom (DOF) needed to characterize the molecular motions. While many molecular simulation techniques are available to generate trajectories to sample the accessible conformational ensemble characterized by those DOF, the investigation of the trajectory may lead to better understanding of how proteins perform biological functions. The process of extracting information from sampled conformations over a trajectory is a task well suited for statistical analysis. Specifically, Principal Component Analysis (PCA) is a multivariate statistical technique applied to reduce the number of dimensions needed to describe protein dynamics through a decomposition process that filters observed motions from the largest to smallest spatial scales (1-5). PCA is a linear transform that extracts the most important elements in the data using a covariance matrix or a correlation matrix (normalized PCA) constructed from coordinates that describe the accessible DOF of the protein, such as the Cartesian coordinates that define atomic displacements in each conformation comprising a trajectory (6). When all of the atomic displacements posses similar standard deviations, a covariance matrix is typically used, otherwise it is prudent to employ the correlation matrix, which normalizes the variables to prevent rare but large atomic displacements from skewing the results. In constructing the covariance matrix or correlation matrix (henceforth C-matrix will be generically used for either matrix type), it is often assumed that the amount of sampling is sufficient, but this always requires many more observations than the number of DOF (variables) used in the matrix. An eigenvalue decomposition (EVD) of the C-matrix leads to a complete set of orthogonal collective modes (eigenvectors), each with a corresponding eigenvalue (variance) that characterizes a portion of the motion, where larger eigenvalues describe motions on larger spatial scales. When the original (centered) data is projected onto an eigenvector, the result is called a principal component (PC).

While PCA can be performed on any high dimensional dataset, for the analysis of a protein trajectory, a C-matrix associated with a selected set of atomic positions must be constructed. Often, a coarse grained description of the protein motion is made at the residue level by using the alpha carbon atom as a representative point for the position of a residue. In this case, the C-matrix will be a  real, symmetric matrix, where  is the number of residues. Performing an EVD results in  eigenvectors (modes) and non-zero corresponding eigenvalues, provided that at least  observations are used. When the eigenvalues are plotted against mode index that are presorted from highest to lowest variance, a “scree plot” typically appears as a function of mode index. When such a scree plot forms, a large portion of the protein motions can be captured with a remarkably small number of modes that define a small dimensional subspace. When analyzing proteins, 20 modes are usually more than enough (even for large proteins) to define an “essential space” that captures the motions governing biological function, thus achieving a tremendous reduction of dimension.

The process of applying PCA to a protein trajectory is called Essential Dynamics (ED) since the “essential” motions are extracted from the set of sampled conformations (8-10). Of course, a linear combination of theorthogonal PCA modes can be used to describe exact protein motions (at the selected coarse grained level). In practice, the presence of large-scale motions makes it difficult or impossible to resolve small-scale motions because the former has much greater relative amplitude in atomic displacements. Indeed, it is for this reason that the large-scale motions are often the most biologically relevant. Therefore, only a small number of PCA modes having the greatest variances are used to characterize large-scale protein motions. When small-scale motions are of interest, the method of PCA can still be used successfully by applying it to sub-regions of a protein as a way to increase the resolution for describing the dynamics within those sub-regions.

Keep in mind that individual PCA mode directions are subject to errors related to finite sampling of conformations to construct the empirical C-matrix. While in theory the empirical C-matrix should be a good estimate for the actual population C-matrix (infinite samples), in practice PCA can be strongly influenced by the presence of outliers in a dataset. The main concern is that any outliers may skew the first few mode directions and steps should be taken to prevent this. As the mode number increases the core part of the essential subspace becomes stable against sampling noise, however only the top several modes tend to be useful. The choice of which modes to include is often made by examining the scree plot for a visible “kink” (the Cattell criterion) (33,34), such that all modes up to the kink are important. Although a kink does not have to exist, it typically does in the study of protein dynamics. In fact, a kink will generally appear for any high dimensional dataset. Hence the name scree (geological debris at the bottom of a cliff) plot has been tied to PCA.

When PCA is applied to Cartesian coordinates that describe the positions of atoms, an alignment step is necessary prior to the process of constructing the C-matrix because the intent is to capture the internal motions of a protein. The structural alignment step requires the center of masses to coincide as well as a global rotation to optimally align the structures. One way to do this is to use a quaternion rotation method to obtain optimal alignment defined by the minimum least-squares error for the displacements between corresponding atoms (35). We note that PCA is not limited to the analysis of a Cartesian coordinate-based C-matrix. Any set of dynamic variables that describe the protein motion can be used. For example, one may choose to use interatomic distances, which eliminates the need to optimally align conformations. In this case, internal atomic distances offer the possibility of an all-to-all distance C-matrix for the alpha carbons, which has a row dimension equal to the number of structures in the trajectory and a column dimension equal to , where  is the number of residues considered. A distance based C-matrix can be created, which is a square matrix with dimension, and therefore requires much more sampling. In this case, the PCA modes reveal the coordinated changes in distances between all residue pairs. Despite the advantage of working directly with internal coordinates, performing all-to-all distance PCA quickly becomes computationally prohibitive due to the need to diagonalize very large non-sparse matrices. More importantly, the interpretation of the eigenvectors becomes difficult when the number of residues is greater than ten. Nevertheless, this approach has proven useful when studying a small subset of atoms where the interpretation is clear (36,37).

For certain applications it is prudent to combine multiple conformational ensembles (CE) together that define a single dataset. One reason for combining different CEs is to boost statistics, where each CE has the same characteristics. This is convenient, as the simplest way to apply parallel computing occurs when multiple simulations are run simultaneously and independently. However, the CEs that are combined could represent different conditions, such as different temperatures in MD simulation, fixing a different set of distance constraints in geometric simulation or contrasting mutant structures. Clustering different CEs in the subspace defined by the most relevant PCA modes provides insight into the effect of varying conditions. A protein may undergo large-scale (anharmonic) conformational changes that bridge two distinct basins of low free energy and the combined CEs will allow these basins to be clearly identified, as well as the paths connecting them. Similarly, different CEs that represent a set of mutant structures, or apo and holo forms of a protein, possibly with different ligands bound, allow one to differentiate the conformations easily by clustering in a small dimensional subspace.

An appealing and intuitive way to investigate protein motions is to project the displacement vectors (DV) defined in the original high dimensional space that characterize different conformations onto a pair of PCA modes. It is even possible to project onto higher dimensions as one visualizes multiple PCA modes simultaneously using specialized software such as R or XL-STAT ™ , which is a plug-in for Microsoft Excel developed by Addinsoft™. Such plots are indispensible for assessing how well certain parts of the subspace are sampled, especially in comparative studies where differentiation in dynamics can have functional consequences. The results of such an analysis show how each state occupies a region of the conformational space defined by the first two PCA modes.

Given that the ED of a protein is well characterized using a small vector space defined by PCA modes that reflect different CEs or perhaps a combined CE, it is desirable to benchmark how similar these subspaces are to one another. When subspaces are sufficiently similar, this implies that the different ensembles capture the same type of protein dynamics. Conversely, when subspaces are very dissimilar, different types of motions are being captured, which may have biological consequences tied to the different conditions analyzed. As such, it is necessary to define a measure to quantify the overlap of vector subspaces, as a natural generalization to the concept of a projection (dot product) of one vector onto another. We note that a set of  PCA modes forms an orthogonal  dimensional *subspace* (SS) within the full *vector space* (VS) defined by the size of the C-matrix. Common metrics that quantify SS similarity include cumulative overlap (CO), root mean square inner product (RMSIP), and principal angles (PA) (12, 41-45). The CO metric quantifies how well one SS is able to capture the PCA modes of the other SS. The RMSIP metric is a single number that quantifies the SS similarity in terms of multiple inner products between the two. The PA method provides a quantification of the optimal alignment between the two SS that is based on the singular value decomposition (SVD) of a matrix of overlaps (inner products) between the two SS. The result is a sorted (monotonically increasing) set of  angles, where  is the dimension of the compared subspaces, that quantify how well the two SS can be aligned.

**Implementation**

At the time of this work, there was no single software package available that would perform all the tasks needed for a complete essential dynamic analysis. JED was designed to perform these tasks in a single software package. JED is written in Java and implements the JAMA Matrix package as well as the Apache Commons Library in order to address each task critical to performing ED in a bioinformatics setting. The JED program performs the following tasks:

1. Allows the user to run multiple jobs using the same set of parameters.
2. Reads in sets of PDB files or coordinate matrix files.
   1. The PDB files may be single chain or multi chain
3. Allows the user to select a set of residues for the analysis that need not be contiguous.
   1. In multi chain PDBs, the residues may come from the various chains.
4. Allows the user to select cPCA, dPCA, both, or none.
5. Allows the user to choose the number of modes to retain for cPCA and dPCA.
6. Allows the user to select cPCA modes for visualizations
   1. For each mode selected, a set of PDB files is generated along with a Pymol script that allows the set of structures to be viewed as a movie. The PDB files have their B-factors replaced by the values of the corresponding MSF PCA mode values.
7. The program performs analysis at the coarse-grain level of all alpha carbons.
8. For the NO PCA option, the following are determined:
   1. The optimal alignment of all trajectory frames.
   2. The matrix of PDB coordinates (original and transformed)
   3. The residue and conformation RMSDs.
   4. A PDB file with the B-factors replaced with the residue RMSDs.
9. For cPCA, the following are also determined:
   1. The displacement vectors (DV).
   2. The covariance (Q) and correlation (R) matrices
   3. The eigenvalues (top and all for Q and R).
   4. The eigenvectors (top for Q and R).
   5. The RMSF PCA modes (normed and weighted; top for Q and R).
   6. The MSF PCA modes (normed and weighted; top for Q and R).
   7. The DV projections onto the top eigenvectors (normed and weighted; Q and R).
10. For dPCA, the following are also determined:
    1. The all-to-all-distances matrix.
    2. The displacement vectors (DV).
    3. The covariance (Q) and correlation (R) matrices
    4. The eigenvalues (top and all for Q and R).
    5. The eigenvectors (top for Q and R).
    6. The weighted eigenvectors (top for Q and R).
    7. The DV projections onto the top eigenvectors (normed and weighted; Q and R).
11. A log file is generated that summarizes the run information. Additionally, when PDB files are read, a PRB\_Read\_Log is also generated listing the names of every file read and the order in which it was read.
12. All outputs are standard .txt files that can be read by programs for additional analysis or graphical viewing.
13. When the program runs, all job information is provided by a file named JED.txt that is read in when the program is run. This file must be in the same directory from which the JVM is called. The syntax for creating the input files is explained in the documentation and is made simple by our web site that has a page for generating and downloading the input files.
14. There are additional tools that are part of the JED package that perform additional tasks once one has generated sets of outputs for each individual trajectory:
    1. Create\_Augmented\_Matrix.java allows one to pool multiple trajectories into one for analysis by JED.
    2. Subspace.java allows one to run comparisons between the individual trajectories and/or a pooled trajectory. The outputs are cumulative overlaps (CO), root mean square inner product (RMSIP), and principal angles (PA) in the log and as test files.
15. The authors maintain a web site where the JED package may be downloaded along with the documentation. Additional resources are provided regarding PCA and essential dynamics as well as contact forms and forms for creating the JED input file.

**Results and Discussion**

The JED software was used to analyze protein trajectories in order to perform a complete essential dynamic analysis. The proteins investigated ranged in size from 100 to 1,000 residues and the number of frames in a trajectory ranged from 2,000 to 100,000. In all cases, the performance was excellent and the most of the processing time was spent calculating the eigenvalue decomposition of the covariance (Q) and correlation matrix (R). The program is moderately intensive in memory due to the fact that all the data for a trajectory is held in memory for the computation of Q (order O2), which is then diagonalized (order O3), and R is computed from Q. The program performs a complete eigenvalue decomposition using the JAMA matrix package. Trajectories derived from myoglobin simulations (153 residues) containing 2,000 frames completed in less than 30 seconds while trajectories derived from myosin (976 residues) containing 2,000 frames completed in about 5 minutes on a MacBook Pro with 8GB ram. It is critical to note that the large datasets and the associated memory challenge of performing ED must be met by using the 64 bit version of JAVA. The 32 bit versions are not able to instantiate VMs with sufficient heap size.

A typical example of how one would use JED is given below:

First, obtain a trajectory. As a best practice, the number of frames should be at least ten times the number of variables (here residues or alpha carbons) in the statistical analysis.

Create the input file JED.txt that is read in by the program when JED is executed. This file must be located in the directory from which the JVM is called. The process of creating the input file is facilitated by provided samples and by using the JED Home page input file generator. The input file provides the data needed to execute the program in a batch mode, using the same parameters for many jobs (or just one), as well as the specific information for each job like the directory of the PDB files and the name of the PDB reference file. The JED program can be called with no command line parameters as all the necessary information is provided by the input file. This makes running JED on computer clusters easy when job scripts need to be written. The output from JED is written to each job directory specified in the input file. All output files are standard flat files (text files) that can be read by any visualization software such as Excel, MatLab, R, GNU Plot, etc. The JED program provides extensive data output that is intended to be visualized and examined to answer questions about the biological function of the protein. Mode visualization is provided my outputting sets of PDB files that can be played as a movie in Pymol and other similar software packages. JED provides a Pymol script for loading the sets of PDB files to show a cartoon view of the backbone colored by the residue rmsd. A key additional set of calculations is performed by the Subspace analysis program that is part of the JED package. This program compares sets of eigenvectors derived from different trajectories where the same protein and same number of residues are subjected to different experimental conditions. This program provides metrics such as CO, RMSIP, and PA to show how similar two trajectories are to each other.

In future releases of the JED package we intend to implement the ability to select various levels of coarse graining from only alpha carbons (current) to all atoms. While using all alpha carbons gives a good idea of the large scale motions of sets of residues, we would like to provide tools for narrowing in on the back bone motions and even on the motions of all heavy atoms to gain insight into how the side chains are behaving. In rare situations, it may be useful to examine all atoms of a small set of residues involved in an active site of the protein. Additionally, we have begun to work with a java 2D chart API called Java Chart2D that creates nice charts and graphs. We intend to include the option to view the generated data files at execution time in Java panels when JED is being run in an interactive way, i.e., not queued on a cluster.

**Conclusions**

We have developed a complete essential dynamics analysis package written in Java that performs all of the tasks that such an analysis requires and does so following best practices in the underlying statistics. The program is written entirely in Java and implements other open source java packages such as JAMA Matrix. The program is accessible to researchers who are not computer programmers and are not well versed in the underlying multivariate statistical mathematics. We have constructed a web site that explains the program operation and provides support for generating the input files that are a key step in using the software. Additionally there is a user manual available for download that provides some example configurations that we used in our testing, and the JED package is available for download (zip and jar files).

**Availability and requirements**

* **Project name:** JED: Java Essential Dynamics
* **Project home page:** http://sourceforge.net/projects/JED
* **Operating system(s):** Platform independent
* **Programming language:** Java
* **Other requirements:** Java JDK 1.5 or higher, an amount RAM appropriate to the size of Q (JED performs a full eigenvalue decomposition).
* **License:** GNU GPL.
* **No restrictions to use:** For reproduction and development, cite thelicense