Protein Motions through Eigenanalyses: A Set of Study Cases

Osni A. Marques and Yves-Henri Sanejouand

CERFACS Report TR/PA/95/32

Protein Motions through Eigenanalyses: A Set of Study Cases

Osni A. Marques[†] Yves-Henri Sanejouand[‡]

October 1995

Abstract

The study of collective motions of molecules provides useful insights into the large amplitude conformational changes the molecules experiment during chemical reactions. In particular, theoretical normal modes analyses of proteins taking into account the lowest-frequency modes may help to predict the nature of such conformational changes. This work lists a set of proteins for which the theoretical motion history has been examined. Focus is given on the computation of low-frequency modes (eigenvalues and eigenvectors) using a code based on the Lanczos algorithm. The normal modes approach and the main ideas governing the technique employed to compute the required modes are first outlined. Then, five distinct cases ranging from 396 to 8528 atoms are discussed. Finally, guidelines for the eigenanalyses of similar problems are proposed.

[†]CERFACS, Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique, 42 av. G. Coriolis, 31057 Toulouse Cedex, France, e-mail: marques@cerfacs.fr

[‡]Laboratoire de Physique Quantique, IRSAMC, Université Paul-Sabatier, 118 route de Narbonne, 31062 Toulouse Cedex, France, e-mail: yves@irsamc1.ups-tlse.fr

1 Introduction

The solutions of

$$Ax = \lambda Bx \tag{1}$$

where A and B are $n \times n$ matrices, x is a non null vector and λ is a scalar, are of great importance in a variety of disciplines. The above relation defines a *generalized eigenproblem*, a *standard eigenproblem* is obtained when B is equal to the identity matrix. In many cases, eigenproblems are associated with fundamental characteristics of differential and integral operators describing a physical phenomenon. The dimensions of A and B can thus reach dimensions of tens of thousands due to the complexity, the level of the discretization of a continuous problem or the precision required for the results. Real applications commonly require the computation of only a small subset of the n *eigenpairs* (λ, x): the *eigenvalues* λ of interest are those lying either in one of the extremities of the spectrum or in an interval $[\xi_1, \xi_2]$, together with the corresponding *eigenvectors* x. However, even the determination of a few solutions of (1) uses to be a time consuming task, justifying a search for efficient algorithms.

Lanczos' [18] and Arnoldi's [2] methods are widely used nowadays for treating eigenproblems associated with large sparse matrices. These techniques generate an appropriate basis of vectors aiming at the projection of the original problem into a smaller problem, involving a tridiagonal matrix (Lanczos) or a Hessenberg matrix (Arnoldi). Eigensolutions are then computed for these reduced problems. Approximate pairs $(\hat{\lambda}, \hat{x})$ for the original problem are obtained by means of a Rayleigh-Ritz or a Galerkin approach. At each step, one vector, or more if a variant by blocks is employed, is added to the basis. Convergence is then achieved as the basis size, m, increases and it usually happens with $m \ll n$. The algorithms of Lanczos and Arnoldi are also attractive because they do not perform modifications on the matrices of the problem. See [6, 14, 24, 25, 26] for a detailed presentation of these methods.

The study of collective motions of molecules provides useful insights into the large amplitude conformational changes the molecules experiment during chemical reactions. Such motions are likely to play an important role in enzymatic activities [4, 10, 23]. In particular, theoretical *normal modes analyses* of proteins[§] taking into account the low-frequency modes help to predict the nature of their conformational changes. As shown in [21] for *citrate synthase*, for example, where conformational transitions involve the relative movement of almost rigid structural elements, theoretical results match those obtained with X-Ray crystallography. The normal modes method consists in obtaining an approximation for the history of motion of the protein through the superposition of collective variables, namely the *normal modes coordinates*. These coordinates are obtained as solutions of a real symmetric eigenproblem whose dimension is equal to three times the number of atoms.

This work lists a set of proteins for which the theoretical study of collective motions has been performed. We focus on the computation of low-frequency modes using a code based on the Lanczos algorithm. The normal modes approach is outlined in the next section. A brief description of the eigenextraction technique is also given. Then, five proteins whose number of atoms vary from 396 to 8528 are examined. We conclude with a discussion on the main issues the practitioner should be aware of when dealing with similar problems.

[§] Using the simplified definition given by Nielsen [23], "proteins are formed by amino acids strung together, like pearls in a bracelet". The amino acids are usually referred to as *residues*.

2 Normal modes approach

In the neibourghood of a stationary point, the potential energy, V, of a system can be approximated by

$$V = \frac{1}{2}\sum_{i=1}^{3n}\sum_{j=1}^{3n} k_{ij}(r_i - r_i^s)(r_j - r_j^s), \tag{2}$$

where the k_{ij}'s are the second derivatives of the potential energy with respect to coordinates r_i and r_j, and r_i^s and r_j^s are the i and j coordinates of the stationary structure. With approximation (2), the equations of motion of the n atoms of the system can be solved analytically by means of

$$r_i(t) = r_i^s + \frac{1}{\sqrt{m_i}}\sum_{j=1}^{3n} a_{ij}q_j(t), \qquad i = 1, 2, ...3n \tag{3}$$

with

$$q_j(t) = C_j \cos(\omega_j t + \phi_j), \tag{4}$$

which means that each atomic motion, $r_i(t)$, results from the superposition of $3n$ independent sinusoidal contributions, or normal modes. In the above equations, m_i is the atomic mass, C_j and ϕ_j are the amplitude and phases, respectively, of normal mode j (dependent upon initial conditions), ω_j is the frequency of normal mode j, obtained as the square root of the j-th eigenvalue of the mass-weighted second derivatives of the potential energy matrix ($\omega = \sqrt{\lambda}$), and the vector $a_j = \{a_{1j}, a_{2j}, \ldots a_{3nj}\}^T$ is the j-th eigenvector of the aforementioned matrix. At a given temperature, the lower the frequency of a normal mode, the larger its amplitude.

3 Computing the normal modes

The normal modes and related frequencies are obtained from the solutions of a real symmetric eigenproblem $Ax = \lambda x$, with

$$A = M^{-1/2}(\nabla^2 E)M^{-1/2}, \tag{5}$$

being E the potential energy matrix, and M the atomic masses matrix (diagonal). The dimension of E and M is three times (the spatial coordinates) the number of atoms. Usually, the normal modes whose frequencies lie under 30–100 cm^{-1} (the smallest eigenvalues) are responsible for most of the amplitude of the atomic displacements of proteins [19, 27]. Since the protein is free in the space, the 6 smallest eigenvalues (and eigenvectors) have no practical interest because they are associated with free (zero energy) molecular motions. Therefore, A is singular and the rigid body motions are not taken into account in the normal modes analysis. The definition of A allows us to devise also the generalized eigenproblem

$$(\nabla^2 E)y = \lambda M y,$$

with $y = M^{-1/2}x$. However, in the CHARMM 21.3 package [5], which has been employed to model the proteins and to compute and minimize the potential energy, data is retrieved as indicated in (5). Moreover, the coordinate system chosen plays a role in the structure of the problem. Cartesian coordinates have been used in our experiments.

In this work, eigenvalues and eigenvectors were computed with the public domain package BLZPACK described in [21]. BLZPACK is a Fortran 77 implementation of the block Lanczos method (in combination with a modified partial reorthogonalization and a selective orthogonalization strategies) for the computation of eigenvalues λ and eigenvectors x of the standard problem $Ax = \lambda x$ or the generalized problem $Ax = \lambda Bx$, where A and B are real symmetric matrices (in this case, a positive definite linear combination of A and B must exist).

The governing idea of the Lanczos algorithm is the generation of an appropriate basis for a Krylov subspace. The Krylov subspace associated with a symmetric matrix A of order n and a starting vector q_1 of unitary length is defined as

$$\mathcal{K}(A, q_1, j) = span(q_1, Aq_1, ...A^{j-1}q_1). \tag{6}$$

The projection of the original problem into the basis leads to a smaller problem, involving a symmetric tridiagonal matrix. Approximate eigenpairs $(\hat{\lambda}, \hat{x})$ for the original problem are then recovered through a Rayleigh-Ritz procedure [25]. Convergence for the dominant eigenvalues of A is usually achieved with $j \ll n$. Associated with the block strategy there is a Krylov subspace built from a full rank $n \times p$ matrix $Q_1 = [\ q_1^{(1)}\ q_2^{(1)}\ ...\ q_p^{(1)}\ ]$, $Q_1^T Q_1 = I$, $1 < p \ll n$, where p is the *block size* [14, 25]:

$$\mathcal{K}(A, Q_1, j) = span(Q_1, AQ_1, ...A^{j-1}Q_1). \tag{7}$$

An approach by blocks allows for better convergence properties when there are many multiple eigenvalues and also a better data management on some computer architectures.

In the general case, the Lanczos algorithm requires calculations involving the matrices A, B and sets of vectors until convergence for the required solutions is reached. However, in the BLZPACK implementation, each time such calculations have to be performed the control is returned to the user, which means that A and B do not have to be passed as arguments for the interface module. Actually, BLZPACK is tailored to a class of applications for which at least one "inversion" of the operator $A_\sigma = A - \sigma B$ is feasible, where σ is a real scalar. With such an inverted operator the eigenvectors are preserved while the eigenvalues are remapped (the Krylov subspaces are then associated with A_σ^{-1} instead of A). Eigenvalues lying in a range of interest can be therefore set apart from the remaining eigenvalues, leading to better converge rates for the wanted solutions. In practical cases, the inversion is replaced by a factorization $A_\sigma = LDL^T$, where L is a lower unit triangular matrix and D is a direct sum of 1×1 and 2×2 pivot blocks, allowing for solutions of systems of linear equations for the basis generation. Since the matrices A and B are kept outside the code, the user is free to employ specific storage strategies or experiment with distinct factorization routines.

As previously discussed, the normal modes coordinates are obtained as solutions of an eigenproblem $Ax = \lambda x$. We recall that the solutions of interest are those in the lower end of the eigenvalue spectrum, thus justifying the use of A_σ^{-1}. For convenience, the user can run the problem as generalized, using $B = I$. In this case, the eigenvalues of A are automatically retrieved by BLZPACK from those of A_σ^{-1}. Alternatively, the problem can be rewritten and solved as $A_\sigma^{-1}x = \theta x$, where $\theta = \frac{1}{\lambda - \sigma}$. With this approach, BLZPACK skips over steps that are normally required for generalized problems and returns the largest θ. The wanted values are therefore simply given by $\sigma + \frac{1}{\theta}$ while the eigenvectors do not need any modification. This was the strategy applied to the experiments described in the next section.

4 Study Cases

In this section, we discuss on the computation of subsets of normal modes of five proteins. Table 1 gives the name of each protein, the dimension, n, of the associated matrix A, and the number of non zero coefficients, nz, in the upper triangle of A. All cases examined here were obtained from the Brookhaven Data Bank [3]. Their representations were drawn with Molscript [17]. In general, we measured the CPU time required (in seconds) and the number of steps performed for the convergence of a given number of eigenpairs, NREIG, varying p and σ (block size for the Lanczos algorithm and translation of origin). However, several eigenvalues can converge simultaneously resulting in more than NREIG solutions available at the end. The first four cases in Table 1 were examined on an IBM Risc 6000/950, using double precision. The largest case was examined on a CRAY C90, using single precision.

The eigenvalues are listed with the associated residual errors $\eta_k = \|\hat{A}\hat{x}_k - \hat{\theta}_k\hat{x}_k\|$, where $\hat{A} = A_\sigma^{-1}$, $\hat{\theta}_k = \frac{1}{\hat{\lambda}_k - \sigma}$, and $(\hat{\theta}_k, \hat{x}_k)$ is the approximate eigenpair computed by BLZPACK. One should note that η_k depends, among others, on the starting vectors, the block size and the number of steps performed. Nevertheless, it can be estimated at a very low cost (see [21] for details). Distinct strategies were used for the factorization $A_\sigma = LDL^T$ and later solutions of systems of equations, thus replacing operations involving A_σ^{-1}.

A collection of subprograms labelled skypack, intended for linear algebra operations with real symmetric matrices stored in a *skyline* (or profile) arrangement has been coded by one of us. For each column of the matrix, the skyline arrangement stores from the first non-zero element to the diagonal element (profile in) or from the diagonal element to the last non-zero (diagonal out) [9]. Good computational performances can be then achieved providing the semibandwidth, sbw[¶], of the matrix is reduced by a *Reverse Cuthill-McKee* (RCM) ordering, for example. Two factorization modules are available in skypack: a standard approach using BLAS 1 kernels[‖] (skypack^1 thereafter) [16], and a partitioned approach using BLAS 2 and 3 kernels (skypack^2 thereafter) [20]. The partitioned scheme requires two auxiliary arrays, one square and other rectangular, to keep data in fast memory as much as possible. For the IBM Risc 6000/950 their dimensions were set to 64 and 64×128, respectively. For the CRAY C90 they were set to 128 and 128×256, respectively. After the factorization has been performed by either module, the solution phase uses BLAS 1 kernels. The subroutines MA27 and MA47, available in the Harwell Subroutine Library [1], were also employed for the factorization of A_σ and solution of systems of equations. The former subroutine uses a sparse variant of Gaussian elimination and a *symmetric minimum degree* (MMD) ordering, while the latter uses a multifrontal Gauss elimination and a combination of MMD and the Markowitz ordering strategy. Ordering schemes aim at reducing the fill-in and the number of operations to be performed during the factorization process, see [8, 9, 13] for details. By choosing appropriate values for σ, A_σ becomes positive definite. Therefore, pivoting to assure numerical stability is not needed and the factorization can be performed more efficiently. Pivoting in MA27 can be suppressed by setting the variable U in common block MA27D to zero after the symbolic manipulations phase (subroutine MA27A). For MA47, the first entry in the control array CNTL have to be set to zero after default values are specified (subroutine MA47I). Conversely, the factorization modules in skypack do not support pivoting.

[¶] For a symmetric matrix, sbw is the smallest integer such that $a_{ij} = 0$ whenever $|i - j| > sbw$.

[‖] BLAS 1: vector-vector products, BLAS 2: matrix-vector products, BLAS 3: matrix-matrix products.

Table 1: Characteristics of the study cases

protein	n	nz
crambine	1188	134217
lysozyme	3795	490602
ras	4986	669060
arabinose	8592	1161360
citrate synthase	25584	3691020

4.1 Crambine

Crambine is a protein found in some seeds. Figure 1 shows its *ball-and-stick* representation. The first 16 eigenvalues of the corresponding matrix A are listed in Table 2. The residuals η_k and $\rho_k = \hat{x}_k^T A \hat{x}_k$, obtained for the case $\sigma = -0.100$ and $p = 3$, are also listed. The products ρ_k, considering five digits in each component of $\hat{x}_k$, were computed only to verify the accuracy of the approximate solutions for this kind of application.** One can see that the eigenvectors associated with the relevant eigenvalues are very well approximated, the others are more sensitive to the (quasi) singularity of the matrix.

Table 3 shows the computational effort, in terms of CPU time, dedicated to skypack, MA27 and MA47, for the factorization of A_σ. The strategy implemented in skypack^2 was at least 14% faster than the other techniques. Table 3 gives also the number of double precision words required during the factorization and used to store the factors D and L:†† the skyline arrangement uses a static data structure while MA27 and MA47 use dynamic data structures. The numbers in parentheses are the ratios between the work space needed and nz, which gives an indication of the fill-in. MA27 and MA47 required larger arrays for handling data than the skyline arrangement. At the end, the factors computed by MA27 resulted slightly better compressed. Figure 2 shows the distribution of the non zero entries in the upper triangle of A as output by CHARMM and after RCM and MMD orderings.

Table 4 lists the CPU time required using skypack, MA27 and MA47, not including the factorization phase, and j, the number of steps performed by BLZPACK, for NREIG=16. In general, the version using MA27 was the most time consuming. Between parentheses are the percentages of the CPU time spent with solutions of systems of equations. Since j may vary slightly for different orderings of A, and therefore $m = p \times j$, the size of the basis computed, the maximum value detected is given. One can see that the convergence improves as σ approaches the first eigenvalue (in other words, η_k becomes smaller with fewer steps), more significantly from $\sigma = -0.100$ to $\sigma = -0.010$. None of the first 6 eigenvalues satisfied the convergence criterion for $\sigma = -0.100$ and $p = 1$, for example. However, by setting the output level of BLZPACK accordingly, eigenvalue approximations that did not satisfy the convergence criterion are also listed. Then, the user may adopt different stratagems to obtain the missing solutions. Guidelines for dealing with such cases are given in [22].

**Since the eigenvectors computed by BLZPACK satisfy $\|\hat{x}\| = 1$, ρ_k is the Rayleigh quotient for $\hat{x}_k$.

††The variables NRLTOT and NRLADU in common block MA27E for MA27, the 6th and 10th entries of array parameter INFO for MA47.

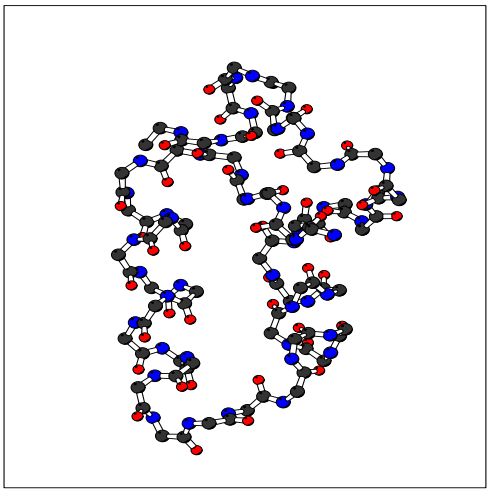


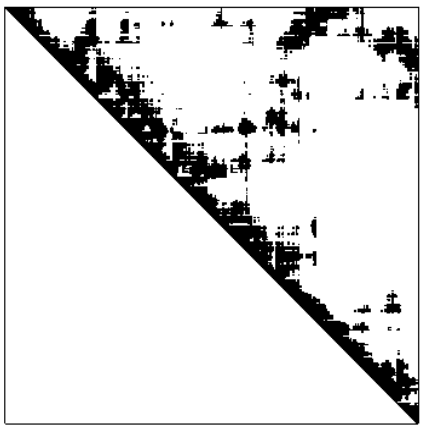
Figure 1: Ball-and-stick representation of *crambine*.

Table 2: Eigenvalues of *crambine*

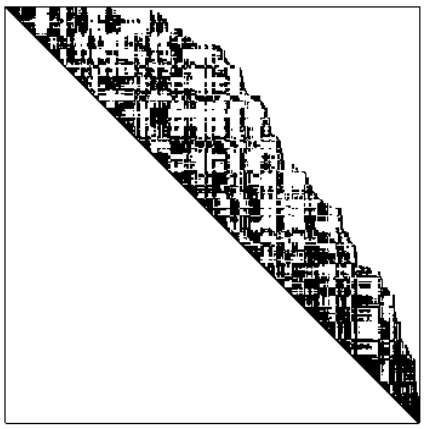
k	$\hat{\lambda}_k$	η_k	ρ_k
1	-7.8839E-09	4.9194E-11	-1.3419E-09
2	-9.3970E-10	2.6122E-09	6.0530E-09
3	-7.5672E-10	3.0656E-09	4.7432E-09
4	1.7218E-10	7.2500E-10	5.1841E-09
5	5.2027E-10	2.0762E-09	8.8761E-09
6	1.8362E-09	1.9175E-09	7.6201E-09
7	2.3736E-03	2.2360E-14	2.3736E-03
8	5.2786E-03	2.3826E-14	5.2786E-03
9	6.7808E-03	4.7708E-14	6.7808E-03
10	1.2590E-02	1.6080E-12	1.2590E-02
11	1.4909E-02	5.1281E-12	1.4909E-02
12	1.6047E-02	1.2831E-11	1.6047E-02
13	1.8084E-02	1.9944E-11	1.8084E-02
14	1.9449E-02	2.0988E-11	1.9449E-02
15	2.3622E-02	1.1872E-10	2.3622E-02
16	2.6567E-02	3.9522E-10	2.6567E-02

Table 3: *Crambine*, factorization of A_σ.

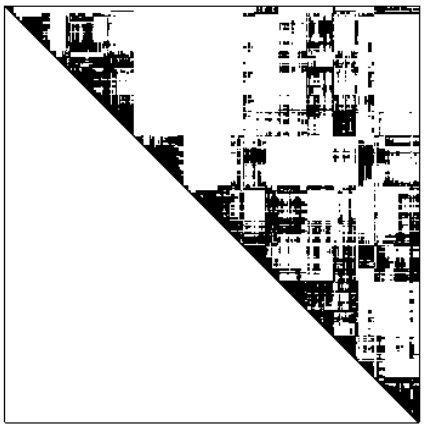
code	CPU	double precision words			
		required		factors	
skypack[1]	6.8	344322	(2.6)	344322	(2.6)
skypack[2]	5.5	344322	(2.6)	344322	(2.6)
MA27	7.9	612085	(4.6)	342927	(2.6)
MA47	6.4	925804	(6.9)	352449	(2.6)



a) CHARMM output.



b) RCM ordering, $sbw = 504$.



c) MMD ordering.

Figure 2: *Crambine*, upper triangle of A.

Table 4: *Crambine*, number of steps and CPU time for NREIG=16.

p	$\sigma = -0.100$				$\sigma = -0.010$				$\sigma = -0.001$			
	j	skypack	MA27	MA47	j	skypack	MA27	MA47	j	skypack	MA27	MA47
1	90	8.3	8.5	7.6	54	3.8	4.2	3.7	43	3.1	3.5	3.0
		(65.9)	(70.0)	(65.8)		(81.1)	(82.0)	(77.9)		(82.2)	(84.0)	(80.5)
2	53	9.1	11.0	9.9	31	4.5	5.1	4.5	28	4.0	4.6	4.0
		(62.3)	(66.8)	(62.7)		(74.3)	(79.3)	(76.7)		(75.3)	(80.2)	(77.1)
3	39	9.3	11.6	10.3	25	5.0	6.5	5.7	23	4.5	5.9	5.2
		(63.6)	(68.5)	(64.9)		(73.9)	(77.7)	(75.2)		(75.4)	(79.4)	(76.2)
4	32	10.5	12.7	11.3	21	5.9	6.9	6.1	19	5.2	6.6	5.8
		(62.2)	(68.7)	(64.7)		(72.2)	(78.7)	(75.6)		(73.5)	(78.8)	(75.8)
5	28	11.8	14.4	12.9	19	7.0	8.0	7.1	17	6.1	7.6	6.8
		(58.9)	(66.1)	(62.2)		(67.9)	(76.1)	(73.1)		(69.7)	(75.8)	(72.2)
6	24	12.0	14.7	13.1	17	7.4	9.3	8.3	16	7.1	8.8	7.8
		(59.2)	(66.6)	(62.8)		(67.3)	(73.4)	(70.1)		(66.7)	(74.0)	(70.4)

4.2 Lysozyme

The *ribbon* representation for the protein *lysozyme* is shown in Figure 3. The first 16 eigenvalues of the corresponding matrix A are listed in Table 5. The residuals η_k and the products ρ_k, for the case $\sigma = -0.0100$ and $p = 1$, are also listed. Again, the products ρ_k, considering five digits in each component of $\hat{x}_k$, were computed only to estimate the accuracy of the approximate solutions. One can see that the eigenvectors corresponding to the significant eigenvalues are very well approximated.

Table 6 shows the computational effort, in terms of CPU time, dedicated to skypack, MA27 and MA47, for the factorization of A_σ. This time, the strategy implemented in skypack^2 was at least 24% faster than the other techniques. Table 6 gives also the number of double precision words necessary for the factorization. The numbers in parentheses are the ratios between the work space needed and nz. MA27 and MA47 required arrays 50% and 114% larger, respectively, for handling data than the skyline strategy. However, the factors they computed were up to 7% better compressed. Figure 4 shows the pattern of the upper triangle of A as output by CHARMM and after RCM and MMD orderings.

Table 7 lists the CPU time required using skypack, MA27 and MA47, not including the factorization phase, and the maximum number of steps performed by BLZPACK, for NREIG=16. In general, the versions using skypack and MA47 had similar performances. The numbers in parentheses are the percentages of the CPU time spent with solutions of systems of equations. Convergence improves as σ approaches $\hat{\lambda}_1$, mainly from $\sigma = -0.100$ to $\sigma = -0.010$. As σ is moved closer to $\hat{\lambda}_1$ the first six eigenvalues are obtained fast, since their counterparts in A_σ^{-1} become fairly isolated (the $\hat{\theta}$-spectrum). However, the relative separation among the interesting values is not improved and therefore the global convergence remains almost unchanged. A similar situation occurs for σ close to $\hat{\lambda}_7$, for instance. An equilibrated convergence rate could be obtained by using a few distinct σ. Nevertheless, since NREIG is not big in this case, that requirement is not mandatory; besides, it would increase the factorization costs.

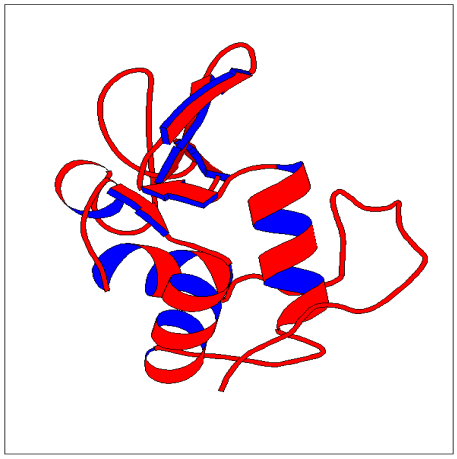


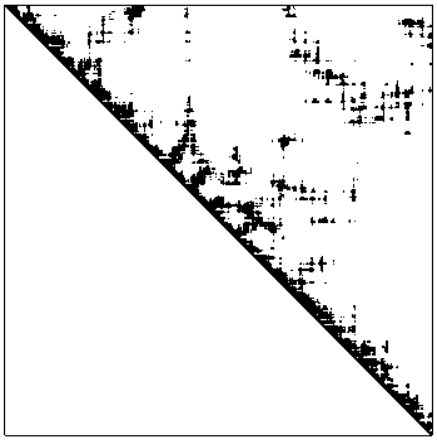
Figure 3: Ribbon representation of *lysozyme*.

Table 5: Eigenvalues of *lysozyme*.

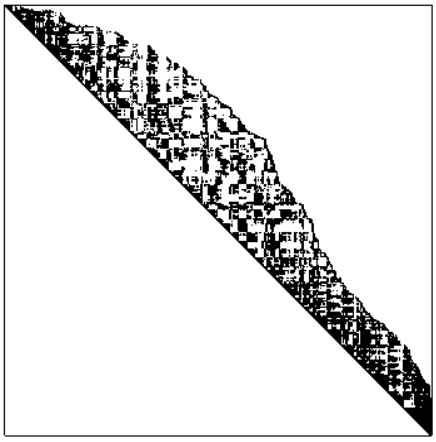
k	$\hat{\lambda}_k$	η_k	ρ_k
1	-9.4463E-07	1.6781E-15	-9.3047E-07
2	-2.8891E-07	9.9311E-14	-2.7744E-07
3	-1.4751E-07	2.8643E-13	-1.3227E-07
4	-7.1575E-09	4.7305E-13	4.3217E-09
5	4.0230E-08	1.2038E-12	5.1140E-08
6	1.5103E-06	1.4811E-16	1.5228E-06
7	1.8665E-03	1.4221E-21	1.8665E-03
8	2.2531E-03	5.0341E-20	2.2531E-03
9	3.0841E-03	1.2265E-16	3.0841E-03
10	3.1389E-03	3.9942E-17	3.1389E-03
11	3.6738E-03	1.9834E-16	3.6738E-03
12	5.0895E-03	3.7787E-14	5.0896E-03
13	5.3663E-03	1.1366E-13	5.3663E-03
14	6.3739E-03	1.1561E-11	6.3739E-03
15	6.9262E-03	1.0013E-10	6.9263E-03
16	7.8606E-03	1.0615E-08	7.8606E-03

Table 6: *Lysozyme*, factorization of A_σ.

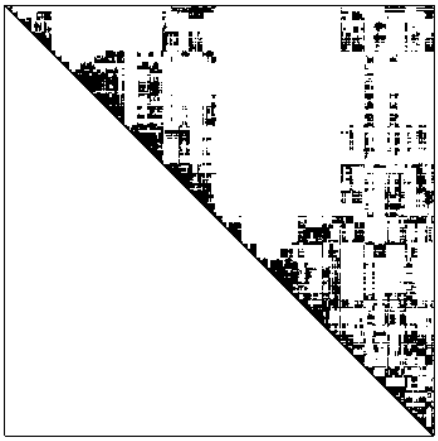
code	CPU	double precision words			
		required		factors	
skypack[1]	79.8	2306568	(4.7)	2306568	(4.7)
skypack[2]	54.6	2306568	(4.7)	2306568	(4.7)
MA27	96.9	3467035	(7.1)	2220078	(4.5)
MA47	71.9	4925977	(10.0)	2156151	(4.4)



a) CHARMM output.



b) RCM ordering, $sbw = 1137$.



c) MMD ordering.

Figure 4: *Lysozyme*, upper triangle of A.

Table 7: *Lysozyme*, number of steps and CPU time for NREIG=16.

p	$\sigma = -0.0100$				$\sigma = -0.0010$				$\sigma = -0.0001$			
	j	skypack	MA27	MA47	j	skypack	MA27	MA47	j	skypack	MA27	MA47
1	62	25.4	27.9	22.1	48	19.4	20.4	16.2	46	18.8	18.8	15.9
		(92.8)	(93.7)	(91.9)		(93.8)	(94.4)	(92.7)		(93.1)	(93.1)	(92.3)
2	39	29.6	36.9	28.9	31	23.1	28.5	23.0	30	22.5	22.5	23.2
		(89.1)	(90.7)	(88.5)		(88.8)	(90.6)	(88.4)		(87.1)	(87.1)	(86.8)
3	30	33.1	42.6	33.6	24	26.6	33.0	26.5	23	25.7	25.7	25.9
		(88.2)	(90.7)	(88.3)		(87.9)	(90.9)	(88.9)		(87.2)	(87.2)	(87.3)
4	26	37.9	48.3	38.6	21	30.3	38.7	31.0	20	29.2	29.2	29.8
		(87.6)	(90.2)	(87.9)		(88.2)	(90.7)	(88.5)		(87.1)	(87.1)	(87.5)
5	23	39.8	54.3	44.1	19	32.5	44.2	35.6	18	33.1	33.1	34.5
		(86.9)	(88.6)	(85.7)		(87.3)	(89.9)	(87.5)		(85.3)	(85.3)	(85.4)
6	21	47.3	56.7	45.6	17	37.9	47.8	38.6	16	36.1	36.1	36.8
		(86.1)	(88.8)	(86.3)		(86.7)	(89.4)	(86.8)		(85.8)	(85.8)	(85.9)

4.3 Ras

Ras is a protein related with transmission of biochemical information between the surface of the cell and its nucleus. Mutations in human *ras* genes are responsible for up to one third of all cases of cancer [7]. The *ribbon* representation for *ras* is shown in Figure 5. Four slightly different models of the protein have been examined. The distribution of the first 50 eigenvalues corresponding to the matrix A of one of the models is given in Figure 6. The values of interest in this case are spread from $\hat{\lambda}_7 = 1.1585 \times 10^{-3}$ to $\hat{\lambda}_{50} = 2.8650 \times 10^{-2}$.

Table 8 shows the computational effort, in terms of CPU time, dedicated to skypack, MA27 and MA47, for the factorization of A_σ. The number of double precision words necessary for the factorization and the corresponding ratios with respect to nz are also given. As for the previous cases, MA27 and MA47 required larger arrays for their data structures (14% and 64%, respectively) than the skyline arrangement. Here, however, the ordering schemes employed in MA27 and MA47 led to much less full-in in the factor L. All the same, these efficient compressings were not reflected in the CPU times, since MA27 was slower and MA47 was only marginally faster than skypack^2 . Figure 7 shows the pattern of the upper triangle of A as output by CHARMM and after RCM and MMD orderings.

Table 9 lists the CPU time required using skypack, MA27 and MA47, not including the factorization phase, and the maximum number of steps performed by BLZPACK, for NREIG=50. The version using MA47 was the fastest and the version using skypack was certainly penalized by the fill-in in the factor L. The numbers in parentheses are the percentages of the CPU time spent with solutions of systems of equations. As can be verified, the convergence rate was not modified by moving σ from -0.0010 to -0.0001 and the CPU time even increased in almost all situations. As σ is moved towards 0 the first six eigenvalues converge fast. In the Lanczos algorithm, convergence is related with a loss of orthogonality among the vectors of the basis generated. If no action is taken, redundant copies of eigenvalues tend to emerge. Thus, additional computations are performed to maintain the orthogonality level under control.

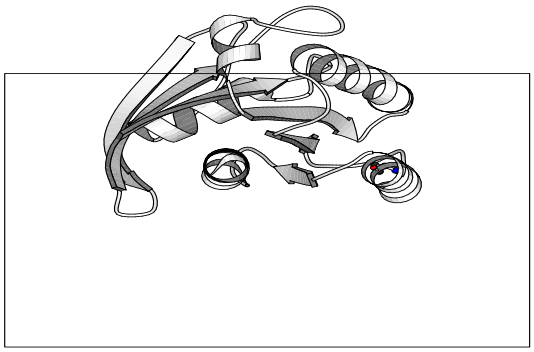


Figure 5: Ribbon representation of *ras*.

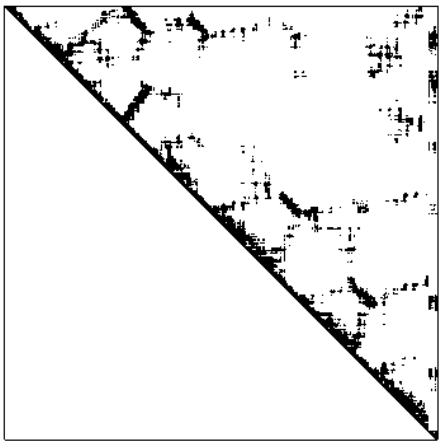
Figure 6: *Ras*, the first 50 eigenvalues ($\hat{\lambda}_1 = -4.4722 \times 10^{-7}$, $\hat{\lambda}_6 = 1.5417 \times 10^{-7}$).

Table 8: *Ras*, factorization of A_σ.

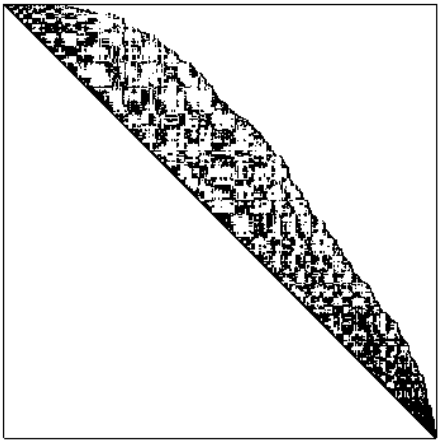
code	CPU	double precision words			
		required		factors	
skypack[1]	205	4490910	(6.7)	4490910	(6.7)
skypack[2]	138	4490910	(6.7)	4490910	(6.7)
MA27	168	5119693	(7.7)	3431601	(5.1)
MA47	132	7355731	(11.0)	3461094	(5.2)

Table 9: *Ras*, number of steps and CPU time for NREIG=50.

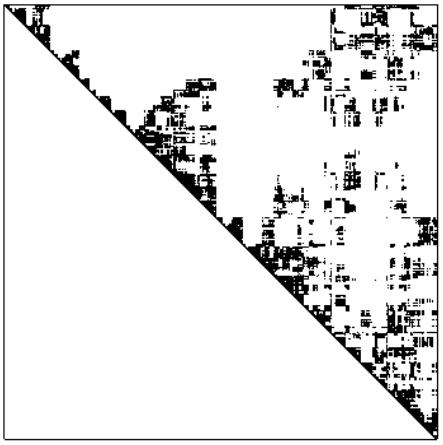
p	$\sigma = -0.0010$				$\sigma = -0.0001$			
	j	skypack	MA27	MA47	j	skypack	MA27	MA47
1	116	96.9	86.1	73.2	114	96.2	87.2	73.1
		(87.3)	(85.6)	(82.8)		(86.0)	(84.1)	(81.4)
2	67	105.	105.	89.3	67	109.	112.	93.5
		(82.8)	(82.5)	(79.0)		(79.8)	(78.8)	(76.1)
3	50	113.	111.	94.8	49	113.	118.	99.8
		(83.8)	(84.1)	(80.9)		(81.9)	(81.7)	(78.1)
4	39	115.	119.	102.	39	118.	124.	108.
		(85.2)	(85.2)	(82.3)		(83.5)	(82.8)	(80.1)
5	35	134.	138.	118.	35	137.	142.	119.
		(83.9)	(83.1)	(79.8)		(81.9)	(81.8)	(78.2)
6	31	150.	143.	123.	31	150.	146.	124.
		(84.1)	(83.1)	(80.1)		(82.8)	(82.1)	(78.6)



a) CHARMM output.



b) RCM ordering, $sbw = 1488$.



c) MMD ordering.

Figure 7: *Ras*, upper triangle of A.

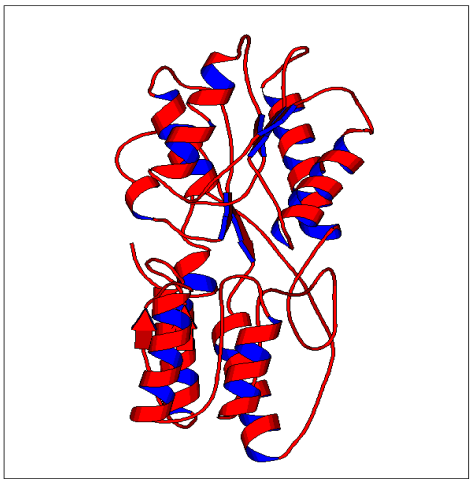


Figure 8: Ribbon representation of *arabinose*.

4.4 Arabinose

The *ribbon* representation for the protein *Arabinose* is shown in Figure 8. The first 16 eigenvalues of the corresponding matrix A, as well as the residuals η_k for the case $p = 3$ and $\sigma = -0.0010$, are listed in Table 10. The magnitudes of the first two negative eigenvalues indicate that additional steps should be performed for a more accurate minimization of the potential energy. As a result of the minimization obtained in this case, the matrix A_σ becomes positive definite with $\sigma \leq -0.0410$.

Table 11 shows the computational effort, in terms of CPU time, dedicated to skypack, MA27 and MA47, for the factorization of A_σ. This time, the strategy implemented in skypack^2 was at least 24% faster than the other techniques. Table 11 gives also the number of double precision words necessary for the factorization. The numbers in parentheses are the ratios between the work space needed and nz. MA27 and MA47 required arrays 22% and 73% larger, respectively, for handling data than the skyline strategy, but the factors they computed were up to 8% better compressed. Figure 9 shows the pattern of the upper triangle of A as output by CHARMM and after RCM and MMD orderings.

Table 12 lists the CPU time required using skypack, MA27 and MA47, not including the factorization phase, and the maximum number of steps performed by BLZPACK, for NREIG=16. As can be seen, the convergence rate was not significantly modified by moving σ from -0.0010 to -0.0001. The versions using skypack^2 and MA47 were competitive in this case, although the L factor was larger in the former. Percentages of the CPU time spent with solutions of systems of equations are given in parentheses. As in the previous cases, moving σ towards the origin did not change the convergence rate considerably. Note that for the values of σ indicated in Table 12 the matrix A_σ becomes indefinite, that is to say, negative values of θ appear. However, if one wishes, the conditioning of the system can be estimated by the approximate eigenvalue spectrum provided by the eigensolver. In addition, errors introduced by an ill-conditioned system would have strong eigenvector components and could be therefore useful in the present application.

Table 10: Eigenvalues of *arabinose*.

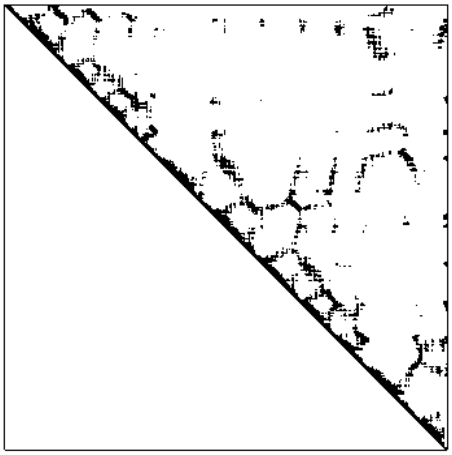
k	$\hat{\lambda}_k$	η_k
1	-4.0968E-02	3.6404E-15
2	-3.1209E-04	9.2563E-15
3	-2.4069E-06	4.0132E-12
4	-1.4992E-06	7.1776E-12
5	1.1459E-07	2.4193E-12
6	1.5875E-06	6.6737E-12
7	2.0016E-06	5.5117E-12
8	6.6040E-06	2.7533E-12
9	4.0125E-04	8.2702E-13
10	8.1116E-04	2.7108E-12
11	1.1388E-03	8.8392E-12
12	1.4323E-03	6.7416E-12
13	1.5169E-03	1.4736E-11
14	2.2458E-03	1.0804E-10
15	2.9136E-03	1.9197E-09
16	3.2549E-03	3.8544E-09

Table 11: *Arabinose*, factorization of A_σ.

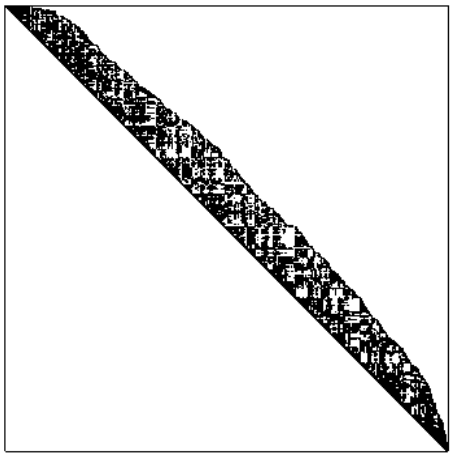
code	CPU	double precision words			
		required		factors	
skypack[1]	365	8089938	(7.0)	8089938	(7.0)
skypack[2]	239	8089938	(7.0)	8089938	(7.0)
MA27	461	9847525	(8.5)	7501134	(6.5)
MA47	337	13992907	(12.0)	7499190	(6.5)

Table 12: *Arabinose*, number of steps and CPU time for NREIG=16.

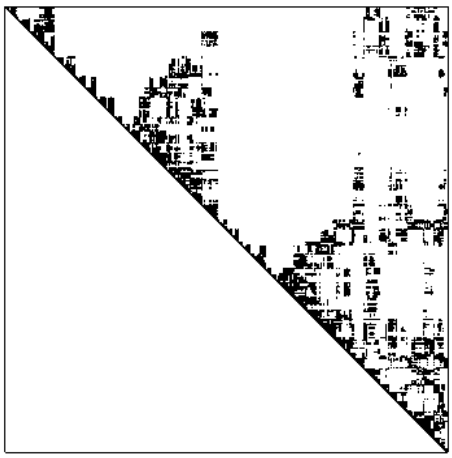
p	$\sigma = -0.0010$				$\sigma = -0.0001$			
	j	skypack	MA27	MA47	j	skypack	MA27	MA47
1	47	64.3	65.2	57.9	45	61.9	61.2	53.8
		(96.0)	(96.2)	(95.7)		(95.6)	(96.0)	(95.6)
2	30	75.1	86.7	77.3	29	74.0	86.0	75.5
		(93.0)	(93.9)	(93.2)		(91.9)	(92.7)	(92.0)
3	25	92.4	104.	92.9	23	85.5	95.9	85.8
		(92.6)	(93.9)	(93.2)		(91.9)	(93.0)	(92.2)
4	22	107.	122.	109.	21	103.	117.	104.
		(92.2)	(93.5)	(92.8)		(91.1)	(92.9)	(92.0)
5	20	124.	148.	131.	19	114.	145.	126.
		(90.7)	(92.3)	(91.5)		(90.2)	(91.8)	(90.8)
6	18	141.	160.	142.	17	138.	151.	134.
		(91.8)	(91.9)	(91.9)		(90.6)	(91.9)	(90.9)



a) CHARMM output.



b) RCM ordering, $sbw = 1395$.



c) MMD ordering.

Figure 9: *Arabinose*, upper triangle of A.

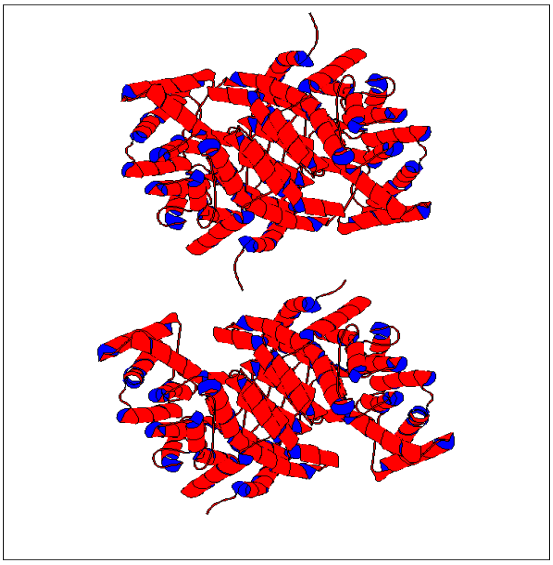


Figure 10: Ribbon representation of *citrate synthase*, closed and open forms.

4.5 Citrate synthase

Citrate synthase is a protein that presents one of the largest *hinge-bending* motions, together with *lysozyme* and *hexokinase*. Theoretical studies of *citrate synthase* are discussed in [11, 12]. Results from a normal modes analysis are given in [22]. The *ribbon* representation of the protein is shown in Figure 10. The first 20 eigenvalues of the corresponding matrix A are listed in Table 13, together with the residuals η_k for $p = 2$. A unique value, -0.001, has been used for σ. In order to avoid underflow, BLZPACK uses a truncation value for small numbers, which explains the residuals equal to 1.0000E-30 in Table 13. Those values indicate a fast convergence for the first six eigenvalues, similarly to the convergence verified for *lysozyme*.

Table 14 shows the computational effort, in terms of CPU time, dedicated to skypack, MA27 and MA47, for the factorization of A_σ, using one processor on a CRAY C90 supercomputer. The number of single precision words necessary for the factorization and the corresponding ratios with respect to nz are also given. All factorization routines required a large work space, up to 24.4 times the original number of non zero entries in A_σ for MA47, that is to say, 719 Mbytes. Moreover, the RCM ordering yielded a matrix with a semibandwidth equal to 4782. However, the ordering schemes in MA27 and MA47 led to much less fill-in in the factor L. The inferior performance of skypack^1 was certainly a heritage of the large fill-in. Still, the performance of skypack^2 suggests that the partitioned strategy is not appropriate for the architecture of the computer employed to solve the problem.

Table 15 lists the CPU time required using skypack, MA27 and MA47, not including the factorization phase, and the maximum number of steps performed by BLZPACK, for NREIG=20. As can be seen, the version using MA47 presented the best performance. The numbers in parentheses are the percentages of the CPU time spent with solutions of systems of equations. Similarly to *arabinose*, such operations were the most time consuming due to the large number of non zero entries in the factors L.

Table 13: Eigenvalues of *citrate synthase*.

k	$\hat{\lambda}_k$	η_k
1	-2.4903E-07	1.0000E-30
2	-1.4109E-07	1.0000E-30
3	-5.8849E-08	1.0000E-30
4	-9.6633E-09	1.0000E-30
5	1.3646E-08	1.0000E-30
6	6.1157E-08	1.0000E-30
7	5.6842E-04	4.0555E-13
8	8.4499E-04	1.1328E-13
9	9.1908E-04	3.4592E-13
10	1.0837E-03	1.0825E-13
11	1.2225E-03	3.4370E-13
12	1.3587E-03	2.5648E-13
13	1.4104E-03	1.0810E-13
14	1.5565E-03	4.1953E-13
15	1.7724E-03	3.8210E-11
16	1.8248E-03	1.5092E-10
17	1.9261E-03	1.8288E-10
18	2.1351E-03	1.8945E-09
19	2.2571E-03	2.4292E-08
20	2.2978E-03	1.8961E-08

Table 14: *Citrate synthase*, factorization of A_σ.

code	CPU	single precision words			
		required		factors	
skypack[1]	603	73037019	(19.8)	73037019	(19.8)
skypack[2]	1964	73037019	(19.8)	73037019	(19.8)
MA27	429	61902628	(16.8)	47430390	(12.9)
MA47	449	89918398	(24.4)	48515655	(13.1)

Table 15: *Citrate synthase*, number of steps and CPU time for NREIG=20.

p	j	skypack		MA27		MA47	
1	65	43.4	(97.9)	32.2	(97.3)	24.3	(96.4)
2	41	51.4	(97.4)	38.0	(97.0)	30.0	(96.0)
3	33	62.8	(97.0)	45.2	(96.3)	36.3	(95.2)
4	27	71.3	(96.9)	49.0	(96.1)	39.5	(94.8)
5	24	78.7	(96.7)	56.8	(95.8)	45.7	(94.4)
6	22	87.1	(96.2)	62.8	(95.3)	50.7	(93.7)

5 Conclusions

This work listed a set of proteins for which the computation of low-frequency normal modes has been performed. These modes are obtained as solutions of an eigenvalue problem and are important for the theoretical study of conformational changes the proteins experiment during chemical reactions. The eigenanalyses were carried out with the public domain package BLZPACK, which is an implementation of the block Lanczos method. The main goal was to give a flavour, to non specialists in eigenanalyses, of the issues involved in the computations, as well as to contribute for an expertise in the study of similar applications.

The formulation adopted in our experiments requires the solution of a a standard eigenvalue problem $Ax = \lambda x$, the pairs (λ, x) of interest being in the lower end of the eigenvalue spectrum. By performing an inversion of the operator $A_\sigma = A - \sigma I$, where σ is a real scalar different from an eigenvalue, the solutions of interest can be obtained efficiently. In practical cases, the inversion is replaced by a factorization $A_\sigma = LDL^T$, using solutions of systems of equations to imitate operations involving A_σ^{-1}. In the BLZPACK implementation, the matrix of the target problem does not need to be passed as an argument for the interface module. This means that each time calculations have to be performed with A_σ, the control is returned to the user. The user is therefore free to employ specific storage strategies or experiment with distinct factorization routines. This flexibility becomes important in many applications, since the factorization usually dominates the computational costs. In this work, three different schemes were used for the factorization of A_σ: a collection of routines intended for linear algebra operations involving real symmetric matrices stored in a skyline arrangement, and the routines MA27 and MA47, available in the Harwell Subroutine Library. For most cases, the dynamic data structures used in MA27 and MA47 required larger arrays than the static data structure employed in the skyline strategy. However, the reordering schemes available in those two routines usually computed factors L with less fill-in than those obtained using a RCM ordering. When the fill-in was roughly the same on the IBM Risc 6000/950, the partitioned skyline approach carried the factorization out in significantly less time, as can be seen in Tables 6 and 11. Therefore, on machines with a cache memory, such a strategy becomes an important option. In addition, appart from RCM, other techniques can be applied to reduce the profile, as the Gibbs-Poole-Stockmeyer algorithm implemented by J. G. Lewis and available in Netlib (algorithm *toms/582*), or those available in the Chaco package [15]. All the same, the operations involving A_σ (factorization and solution of systems of equations) are likely to dominate the computational costs. *Citrate synthase*, for instance, looks like an extravagant case. Therefore, for large problems, we are tempted to think about iterative methods for solving systems of equations, thus avoiding the factorization of A_σ. In this particular, BLZPACK has already been used with success for nonlinear iterations in self-consistent field computations.

Concerning the number of steps performed for the determination of the number of eigenvalues and eigenvectors we have asked for, and in spite of the existence of solutions very close to each other in all problems, we have seen that a block size equal to one led to good performances. If the solvers employed dealt with multiple right hand sides in a more efficient way, the situation could be probably changed. However, our results also showed that as the block sizes increased more vectors were needed in the bases for attaining convergence. The exception to this was verified when σ was applied relatively far from the first eigenvalue, as seen in *crambine*.

Finally, since in theory A is singular, $\sigma = 0$ was avoided. However, we saw that A can be moved from the singularity due to roundoff errors or inaccuracy in the minimization of the potential energy. Depending on the value of σ, it can happen that some of the smallest eigenvalues do not satisfy the convergence criterion. By setting the output level of BLZPACK accordingly, eigenvalue approximations that do not satisfy the convergence criterion are also listed. Then, the user may adopt the stratagems suggested in [21] to obtain the missing solutions. As σ is moved towards λ_1, the first six eigenvalues converge fast because their counterparts in A_σ^{-1} become fairly isolated from the others. Conversely, the relative separation among the interesting values may be not improved and therefore the global convergence may remain almost unchanged. Our experiments indicated that a value for σ equal to -0.0010 or -0.0001 is a good choice. As an alternative, one can solve the problem as generalized, using $B = I$ and asking BLZPACK to perform an automatic spectrum slicing. Nevertheless, since the normal modes analyses are usually carried out with a relatively small number of pairs (λ, x), such an approach seems not to be necessary. In addition, the automatic spectrum slicing requires additional factorizations, which would increase the computational costs.

References

[1] AEA Technology, Didcot, England. *Harwell Subroutine Library.* Release 11 (July 1993).

[2] W. E. Arnoldi. The Principle of Minimized Iterations in the Solution of the Matrix Eigenvalue Problem. *Quarterly of Applied Mathematics*, IX:17–29, 1951.

[3] F. C. Bernstein, T. F. Koetzle, G. J. B. Williams, E. F. Meyer, M. D. Brice, J. R. Rodgers, O. Kennard, T. Shimanouchi, and M. Tasumi. The protein data bank: a computer-based archival file for macromolecular structures. *J. Mol. Biol.*, 112:535–542, 1977.

[4] B. Brooks and M. Karplus. Normal Modes for Specific Motions of Macromolecules: Application to the Hinge-Bending Mode of Lysozyme. *Proc. Natl. Acad. Sci. USA*, 82:4995–4999, 1985.

[5] B. R. Brooks, R. E. Bruccoleri, B. D. Olafson, D. J. States, S. Swaminathan, and M. Karplus. CHARMM: A program for macromolecular energy, minimization, and dynamics calculations. *J. Comp. Chem.*, 4:187–217, 1983.

[6] F. Chatelin. *Valeurs Propres de Matrices.* Masson, Paris, France, 1988.

[7] S. Day. Just Obeying Orders. *New Scientist*, 27 May:26–30, 1995.

[8] J. J. Dongarra, I. S. Duff, D. C. Sorensen, and H. A. Van der Vorst. *Solving Linear Systems on Vector and Shared Memory Computers.* SIAM, Philadelphia, USA, 1991.

[9] I. S. Duff, A. M. Erisman, and J. K. Reid. *Direct Methods for Sparse Matrices.* Clarendon Press, Oxford, England, 1986.

[10] P. Durand, G. Trinquier, and Y.-H. Sanejouand. A New Approach for Determining Low-Frequency Normal Modes in Macromolecules. *Biopolymers*, 34:759–771, 1994.

[11] M. A. Ech-Cherif El-Kettani and J. Durup. Theoretical Determination of Conformational Paths in Citrate Synthase. *Biopolymers*, 32:561–574, 1992.

[12] M. A. Ech-Cherif El-Kettani, K. Zakrzewska, and J. Durup. An Analysis of the Conformational Paths of Citrate Synthase. *Proteins*, 16:393–407, 1993.

[13] A. George and J. W. H. Liu. *Computer Solution of Large Sparse Symmetric Positive Definite Systems*. Prentice Hall, Englewood Cliffs, USA, 1981.

[14] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, USA, third edition, 1996.

[15] B. Hendrickson and R. Leland. The Chaco User's Guide. Version 1.0. Technical Report SAND93-2339, Sandia National Laboratories, Albuquerque, USA, 1993.

[16] T. J. R. Hughes. *The Finite Element Method*. Prentice Hall International Editions, 1987.

[17] P. J. Kraulis. MOLSCRIPT: a program to produce both detailed and schematic plots of protein structures. *J. Appl. Cryst.*, 24:946–950, 1991.

[18] C. Lanczos. An Iteration Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operators. *J. of Res. of the Nat. Bur. of Stand.*, 45:255–282, 1950.

[19] R. M. Levy, D. Perahia, and M. Karplus. Molecular dynamics of an α-helical polypeptide: temperature dependance and deviation from harmonic behavior. *Proc. Natl. Acad. Sci. USA*, 79:1346–1350, 1982.

[20] O. A. Marques. A Partitioned Skyline LDL^T Factorization. Technical Report TR/PA/93/53, CERFACS, Toulouse, France, 1993.

[21] O. A. Marques. BLZPACK: Description and User's Guide. Technical Report TR/PA/95/30, CERFACS, Toulouse, France, 1995.

[22] O. A. Marques. Eigensolvers and Applications in Finite Element Analyses. In M. Papadrakakis and G. Bugeda, editors, *Advanced Solution Procedures on Innovative Computer Architectures*, pages 66–79. CIMNE Publications, Barcelona, 1996. Also CERFACS Report TR/PA/95/29, Toulouse, France.

[23] R. H. Nielsen. Life, The Movie. *New Scientist*, 29 April:32–35, 1995.

[24] B. Nour-Omid. The Lanczos Algorithm for Solution of Large Generalized Eigenproblem. In T. J. R. Hughes, editor, *The Finite Element Method*, pages 582–630, Englewood Cliffs, USA, 1987. Prentice Hall International Editions.

[25] B. N. Parlett. *The Symmetric Eigenvalue Problem*. SIAM (Classics in Applied Mathematics), Philadelphia, USA, 1998.

[26] Y. Saad. *Numerical Methods for Large Eigenvalue Problems*. Manchester University Press, Manchester, England, 1992.

[27] S. Swaminathan, T. Ichiye, W. F. Van Gunsteren, and M. Karplus. Time dependance of atomic fluctuations in proteins: analysis of local and collective motions in bovine pancreatic trypsin inhibitor. *Biochemistry*, 21:5230–5241, 1982.