Lagrange multipliers

Consider an optimization problem with a constraint. For example, we wish to minimize a function \( f(x, y, z) = x^2 + y^2 + z^2 \) with respect to the variables \( x, y, z \). The global minimum is clearly when \((x = 0, y = 0, z = 0)\).

However we now add a constraint \( \sigma = x + y + z - 1 = 0 \).

How can we solve this problem? One approach is to use the constraint(s) to solve for one of the variables. For example, we can use \( z = 1 - x - y \) and minimize the new function \( f(x, y, z) = x^2 + y^2 + (1 - x - y)^2 \).

The difficulty with the above (which is nevertheless an exact procedure) is that it is not always easy to solve explicitly for one of the variables. Moreover, the problem is treated in a very asymmetric way, who said that we should substitute \( z \)? Perhaps simpler equations are obtained when different variable is chosen.

The Lagrange multipliers approach makes it possible to do the constrained optimization in a symmetric way almost like an optimization without constraints. We consider a new function \( g(x, y, z, \lambda) \) with one more variable, \( \lambda \) compared to three variables of \( f(x, y, z) \). The new function is given by

\[
g(x, y, z, \lambda) = f(x, y, z) + \lambda \sigma(x, y, z)
\]

If we now wish to determine a stationary point of \( g(x, y, z, \lambda) \), first derivatives are useful

\[
\frac{dg}{dx} = \frac{df}{dx} + \lambda \frac{d\sigma}{dx} = 0
\]

\[
\frac{dg}{dy} = \frac{df}{dy} + \lambda \frac{d\sigma}{dy} = 0
\]

\[
\frac{dg}{dz} = \frac{df}{dz} + \lambda \frac{d\sigma}{dz} = 0
\]

\[
\frac{dg}{d\lambda} = \sigma = 0
\]

Note that the constraint is a constant function. So in principle, if the last equation holds all the derivatives of the constraint should vanish and we are left with the original constraint minimization. However, we translate the problem into more symmetric and simpler one, direct minimization with one more variable.
Structural Overlap

The motivation behind structural alignments (a protein structure is aligned against another structure) is that this is the best alignment we could possibly get. Since function of proteins is determined by the contacts in three dimensions, the shapes of two related proteins better be similar. Positions of related amino acids are better determinants of the proximity of the two proteins. For example for the two shapes below

We should be able to detect the similarity between the fragments (helices), and point to the loop as the structural segment that deviates the most, regardless of the sequence. Since the identities of the amino acids are not used (only the $C_\alpha$ positions) highly remote evolutionary connections may be observed. This is our final goal. We start however with the (much) simpler case of overlapping proteins of the same length (no alignment is necessary just proper measure of their distance).

Computing the distance between protein structures

We consider two proteins $A$ and $B$ with the same number of amino acids $n$ (the question of alignment of two structures with different number of amino acids will follow the simpler case of overlap). The coordinate vectors of protein $A$ and $B$ are denoted by $X_A$ and $X_B$ respectively. Each of these vectors is of length $3n$ including the (x,y,z) (Cartesian) positions of the $C_\alpha$-s of the amino acids. The rank 3 vector of amino acid $i$ in structure $A$ is denoted by $r_i^A$. The distance between the two structures $D$ is defined (and written explicitly as)

$$D^2 = \sum_{i=1}^{n} \left( r_i^A - r_i^B \right)^2$$

Hence we think on the two proteins as a collection of points, or alternatively as a point in $3n$ space for which we compute norm two of the vector difference $\|X_A - X_B\|_2$. 
Since the coordinates are defined in Cartesian space, it is possible to translate or rotate one of the structures with respect to the other without changing any of the internal distances between the points that belong to the same object, the protein. That is, maintaining its rigid shape. For simplicity we will always move structure $A$.

We will consider the translation and the rotation separately. A translation is defined by adding to each of the $r^A_i$ vector a single constant vector $t$. A rotation is defined by multiplying a coordinate vector by a 3x3 matrix $U$ (e.g. $Ur^A_i$). $U$ satisfies $UU^T = 1$ and $\det(U) = 1$ (** What are the conditions on $U$ good for?**)

Let us start with the simpler problem, that of translation. We wish to determine $t$ so that $D^2$ is minimal. This is trivial

$$D^2 = \sum_{n=1}^{N} (r^A_n + t - r^B_n)^2 = \text{minimum}$$

$$2 \frac{dD}{dt} = 2 \sum_{n=1}^{N} (r^A_n + t - r^B_n) = 0$$

$$t = \frac{1}{N} \sum_{n=1}^{N} (r^B_n - r^A_n) \quad \eta = x,y,z$$

Hence, all we need to do is to correct the position of $r^A_i$ by the difference in the geometric centers of the two proteins. After doing this we will be ready to consider the more interesting problem of overlapping two structures, the problem of rotation.

In fact, to make sure that the next item on the agenda is pure rotation we will set the two geometric centers of the two proteins to zero. In the following derivation we assume that this was already done. We will keep the same notation of $r^A_i$ and $r^B_i$ for the vectors with the adjusted translation.

To correct for possible rotations we write yet another optimization problem

$$D^2 = \sum_{n=1}^{N} (Ur^A_n - r^B_n)^2 = \text{minimum}$$

subject to the constraints: $UU^T = 1$

or $\sum_{k=1}^{3} u_{\eta k} u_{\eta j} - \delta_{\eta j} = 0$

The constraint is inserted to the optimization using Lagrange’s multipliers.
\[ F = D^2 + \sum_{i,j} \Lambda_{ij} \left( \sum_k u_{ik} u_{kj} - \delta_{ij} \right) \]

The unknowns that we wish to determine are all the elements of the \( U \) matrix (9 in all). However, the constraints reduce the number of unknowns (** to how many??**). To find the minimum of \( D^2 \) subject to the constraint of unitary matrix \( U \), we differentiate with respect to the matrix element \( u_{ij} \), we have

\[
\frac{\partial F}{\partial u_{ij}} = \sum_k u_{ik} \left( \sum_n r_{nk} r_{nj}^A + \lambda_{ij} \right) - \sum_n r_{n}^A r_{n}^B = 0
\]

We now define two matrices

\[
R_{ij} = \sum_{n} r_{ni}^B r_{nj}^A, \quad S_{ij} = \sum_{n} r_{ni}^A r_{nj}^A
\]

With the help of the above definition we can write \( \frac{\partial F}{\partial u_{ij}} \) in a more compact form

\[
U (S + \Lambda) = R
\]

We have one matrix equation with two unknown matrices (!) -- \( U \) and \( \Lambda \). Of course, things are not so bad since we still have the constraint equation: \( UU' = 1 \)

Note also that \( (S + \Lambda) \) is a symmetric matrix. On the other hand \( R \) is not symmetric which makes our problem a little more interesting. The following trick will eliminate some of our problems: Multiply the last equation by its transpose:

\[
(S + \Lambda)' U' U (S + \Lambda) = R'R
\]

and using \( U'U = 1 \) constraint eliminates \( U \) from the equation.

\[
(S + \Lambda) (S + \Lambda) = R'R
\]

The eigenvectors of \( (S + \Lambda) \) - \( a_k \) are the same as the eigenvectors of \( R'R \) (assuming no degeneracy). The eigenvalues of \( R'R \) are \( \mu_k^2 \). The corresponding eigenvalues of \( (S + \Lambda) \) are therefore

\[
(S + \Lambda) a_k = \pm \mu_k a_k \quad \text{(the eigenvalues of the square of the matrix are determined only up to a sign)}
\]

Recovering now the original equation we realize that
\[
U (S + \Lambda) = R \rightarrow R_{ij} = \sum_{k} b_{ki} (\pm \mu_k) a_{ij} \rightarrow \\
u_{ij} = \sum_{k} b_{ki} a_{ij}
\]

The set of orthonormal vectors \( b_k \) are obtained by rotating the set \( a_k \) with the (unknown) \( U \). However the \( b_k \) are also the “left” eigenvectors of \( R \). The right and the left eigenvectors, and the eigenvalues can be obtained directly from Singular Value Decomposition (SVD) of the asymmetric matrix \( R \). Finally our optimal distance can be computed more directly without thinking on \( U \) at all (of course to make a nice plot of overlapping structures requires the rotation matrix):

\[
D^2 = \sum_{n} (Ur_n^A - r_n^B)^2 = \sum_{n} (r_n^A)^2 + (r_n^B)^2 - 2 \sum_{n} r_n^B (Ur_n^A) \\
= \sum_{n} (r_n^A)^2 + (r_n^B)^2 - 2 \sum_{n} \sum_{k} (b_k r_n^B) (r_n^A a_k) \\
= \sum_{n} (r_n^A)^2 + (r_n^B)^2 - 2 \sum_{k} (b_k) (Ra_k) \\
= \sum_{n} (r_n^A)^2 + (r_n^B)^2 - 2 \sum_{k} \pm \mu_k
\]

Problems to think about

- What shall we do if the determinant of \( U \) is not +1, or what to do about the sign of \( \mu \)?
- What may happen if both molecules are planar/linear?