$$\hat{d}_Y(y, y') = u^1 d_1 + u^2 d_2 + u^3 d_3 \;=\; d^{\mathrm{T}} u, \tag{9.7}$$

where $d = (d_1, d_2, d_3)^{\mathrm{T}}$.

Let us now remove the restriction that y' coincides with one of the vertices of Y_M and consider the general case where $y' = (t', u')$ is an arbitrary point on the mesh. The major difference is that now the distances $d_1 = d_Y(y_1, y')$, $d_2 = d_Y(y_2, y')$, and $d_3 = d_Y(y_3, y')$ are unknown. Yet, assuming that the triangle t' to which y' belongs is formed by the vertices y_4, y_5, and y_6, we may apply the previously described interpolation method in the triangle t with $d = (d_{14}, d_{24}, d_{34})^{\mathrm{T}}$, $d_{ij} = d_Y(y_i, y_j)$, to obtain $\hat{d}_4 \approx d(y, y_4)$. In a similar manner, $\hat{d}_5 \approx d(y, y_5)$ and $\hat{d}_6 \approx d(y, y_6)$ are obtained. This can be expressed as

$$\begin{pmatrix} \hat{d}_4 \\ \hat{d}_5 \\ \hat{d}_6 \end{pmatrix} = \begin{pmatrix} d_{14} & d_{24} & d_{34} \\ d_{15} & d_{25} & d_{35} \\ d_{16} & d_{26} & d_{36} \end{pmatrix} u \;=\; D_Y(t, t')u, \tag{9.8}$$

where the matrix $D_Y(t, t')$ depends on the triangle indices t and t' only. Now, we may apply the interpolation once again, this time in the triangle t' with $d = (\hat{d}_4, \hat{d}_5, \hat{d}_6)^{\mathrm{T}}$, obtaining the sought interpolant

$$\hat{d}_Y(u, u') = u'^{\mathrm{T}} D_Y(t, t')u. \tag{9.9}$$

This final step completes the picture: now we have a computational tool for the interpolation of the geodesic distances on $T(Y_M)$. Figure 9.6 visualizes the four steps. We leave as an exercise to the reader (Problem 9.9) the proof of the fact that $\hat{d}_Y$ satisfies the five properties stated in the beginning of the section.

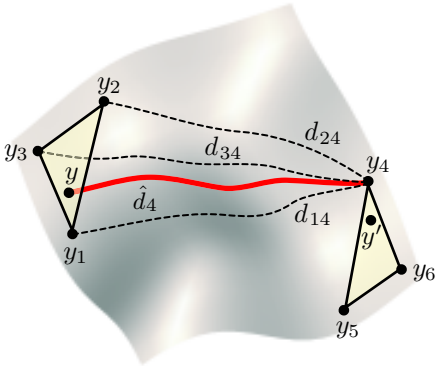
9.5 Minimization of the generalized stress

Substituting the interpolated distance terms $\hat{d}_Y(u_i, u_j) = u_i^{\mathrm{T}} D_Y(t_i, t_j)u_j$ to the generalized L_p stress function, we obtain

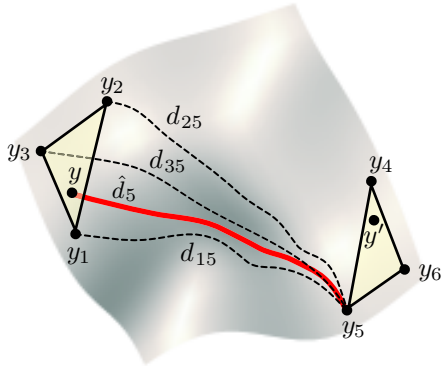
$$\sigma_p^p(t_1, u_1, \ldots, t_N, u_N) = \sum_{j>i} \left(d_X(x_i, x_j) - u_i^{\mathrm{T}} D_Y(t_i, t_j)u_j \right)^p. \tag{9.10}$$

We now have all the ingredients ready for constructing a numerical scheme for the solution of the GMDS problem. In what follows, we will focus our attention on the L_2 case only; other cases can be addressed, for example, by using the iterative reweighting scheme described in Chapter 7. To simplify notation, we will write σ instead of σ_2^2.

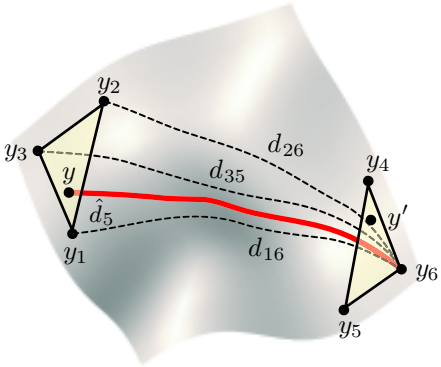
The choice of the L_2 stress has a significant advantage: Because the summation excludes the cases where $i = j$, the interpolated distance terms



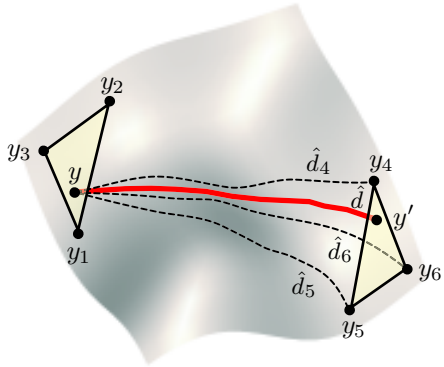
Step 1: $\hat{d}_4 \approx d_Y(y, y_4)$ is computed from d_{14}, d_{24}, d_{34} in the triangle y_1, y_2, y_3.



Step 2: $\hat{d}_5 \approx d_Y(y, y_5)$ is computed from d_{15}, d_{25}, d_{35} in the triangle y_1, y_2, y_3.



Step 3: $\hat{d}_6 \approx d_Y(y, y_6)$ is computed from d_{16}, d_{26}, d_{36} in the triangle y_1, y_2, y_3.



Step 4: $\hat{d} \approx d_Y(y, y')$ is computed from $\hat{d}_4, \hat{d}_5, \hat{d}_6$ in the triangle y_4, y_5, y_6.

Figure 9.6. Four steps performed to interpolate the geodesic distance $\hat{d}_Y(y, y')$ between two arbitrary points on the mesh Y.

$u_i^{\mathrm{T}} D_Y(t_i, t_j) u_j$ are linear in u_i. This implies that fixing all the t_j's and all the u_j's except for some u_i, we obtain a *quadratic* stress function

$$\sigma(u_i) = \sum_{j \neq i} \left(d_X(x_i, x_j) - u_i^{\mathrm{T}} D_Y(t_i, t_j) u_j \right)^2, \tag{9.11}$$

which we write for brevity as $\sigma(u_i) = u_i^{\mathrm{T}} A_i u_i + 2 b_i^{\mathrm{T}} u_i + c_i$, denoting by

$$A_i = \sum_{j \neq i} D_Y(t_i, t_j) u_j u_j^{\mathrm{T}} D_Y(t_i, t_j)^{\mathrm{T}}, \tag{9.12}$$

$$b_i = - \sum_{j \neq i} d_X(x_i, x_j) D_Y(t_i, t_j) u_j, \tag{9.13}$$

and

$$c_i = \sum_{j \neq i} d_X^2(x_i, x_j) \tag{9.14}$$

the quadratic form coefficients. We leave to the reader to show that the matrix A_i is positive semi-definite, that is, $\sigma(u_i)$ is convex.[3]

The quadratic form of $\sigma(u_i)$ allows us to express its minimizer in a closed form,

$$u_i^* = \arg\min_{u_i} \sigma(u_i) \;=\; -A_i^{-1} b_i. \tag{9.15}$$

However, such a solution may happen to be outside the triangle, rendering the barycentric representation invalid. In order to obtain a valid point, we constrain u_i^* to lie within the triangle:

$$u_i^* = \arg\min_{u_i} \sigma(u_i) \quad \text{s.t.} \quad \begin{cases} u_i \geq 0, \\ u_i^1 + u_i^2 + u_i^3 = 1. \end{cases} \tag{9.16}$$

This linearly constrained quadratic problem still has a closed-form solution, the derivation of which is left as an exercise to the reader.

Observe that replacing u_i with u_i^* results in a decrease of the stress value. We can use this fact to construct an algorithm that fixes all the variables except for some u_i, updates u_i with u_i^*, then picks a different u_i while fixing the rest of the variables, and continues in the same way, producing a sequence of updates. A reasonable choice of the point to update can be based on the gradient of the stress function with respect to each u_i,

$$g_i = \nabla\sigma(u_i) \;=\; 2A_i u_i + 2b_i. \tag{9.17}$$

It is desirable to select the u_i with the largest gradient in order to obtain the steepest decrease in the stress function value when the point is updated. The resulting algorithm is similar in its spirit to the coordinate descent (L_1 steepest descent) method from Chapter 5, where each time only one coordinate corresponding with the largest derivative was updated. Here, we operate on a single point u_i, resulting in an update of a block of two coordinates, u_i^1 and u_i^2, in the vector of optimization variables. For this reason, our minimization algorithm belongs to the family of *block coordinate descent* algorithms.

It is important to note that the updated point u_i^* may happen to lie on an edge or a vertex of the triangle t_i (that is, at least one constraint in (9.16) is active). In this case, the need to update the triangle index t_i may arise. If u_i^* lies on a triangle edge shared with some other triangle t_i', we translate the vector of barycentric coordinates u_i^* in the coordinate system of t_i to another vector $u_i'^*$ in the coordinate system of t_i'. This translation does not change the value of $\sigma(u_i)$, yet, as the stress function is not $\mathcal{C}^1$ on the triangle boundaries, the gradient direction may change. We evaluate the new gradient direction in the triangle t_i' and update t_i to be t_i' only if the negative gradient

for $k = 0, 1, 2, \ldots$ **do**
  Compute the parameters A_i, b_i and the gradients g_i of the stress function $\sigma(t_1^{(k)}, u_1^{(k)}, \ldots, t_N^{(k)}, u_N^{(k)})$.
  Select i corresponding to $\max \|g_i\|$.
  if $\|g_i\|$ *is sufficiently small* **then** stop
  Solve the constrained quadratic problem

$$u_i^* = \arg\min_{u_i \geq 0} \sigma(u_i) \quad \text{s.t.} \quad u_i^1 + u_i^2 + u_i^3 = 1$$

  with the rest of the u_j's fixed to $u_j^{(k)}$.
  if u_i^* *is on an edge of the triangle* t_i **then**
    let $\mathcal{N}(u_i^*, t_i)$ be the triangle sharing the edge with t_i, or $\emptyset$ if the edge is on the shape boundary.
  else if u_i^* *is on a vertex of* t_i **then**
    let $\mathcal{N}(u_i^*, t_i)$ be the set of triangles sharing the vertex with t_i.
  else
    set $\mathcal{N}(u_i^*, t_i) = \emptyset$.
  if $\mathcal{N}(u_i^*, t_i) = \emptyset$ **then** update $u^{(k+1)} = u_i^*$, and go to Step 2.
  forall $t' \in \mathcal{N}(u_i^*, t_i)$ **do**
    Translate u_i^* in the triangle t_i to $u_i'^*$ in the triangle t'.
    Evaluate the gradient $g_i = \nabla_{u_i} \sigma(t', u_i'^*)$.
    if $-g_i$ *is directed inside the triangle* t' **then**
      update $t_i^{(k+1)} = t'$, $u^{(k+1)} = u_i'^*$, and go to Step 2.
    end
  end
end

Algorithm 9.1. Least squares GMDS.

direction points inside t_i'. In this case, subsequent minimization of σ with respect to u_i will guarantee a further decrease of the stress function value. If the triangle edge is not shared with another triangle (i.e., the edge is part of the mesh boundary), no index update is performed. The case where u_i^* lies on a triangle vertex is treated in a similar way. The entire minimization procedure is summarized in Algorithm 9.1.

It is worthwhile noting that the described algorithm is also suitable when the surface Y admits a global parameterization. In this case, the point coordinates u_i are expressed in the global system of coordinates, and the triangle indices t_i are inferred from the point locations. As a concluding remark before we proceed to the next topic, we should mention that Algorithm 9.1 is by no means the only way to solve the GMDS problem. For example, in [67] we proposed a different numerical scheme based on a *path search* – a generalization of line search, where instead of a line in a high-dimensional Euclidean space the minimum is searched over poly-linear paths on the mesh $T(Y_M)$.