10.4 Downhill Simplex Method in Multidimensions

With this section we begin consideration of multidimensional minimization, that is, finding the minimum of a function of more than one independent variable.
This section stands apart from those which follow, however: All of the algorithms after this section will make explicit use of the one-dimensional minimization algorithms of §10.1, §10.2, or §10.3 as a part of their computational strategy.
This section implements an entirely self-contained strategy, in which one-dimensional minimization does not figure.
This "downhill simplex method" is due to Nelder and Mead (1965). The method requires only function evaluations, not derivatives. It is not very efficient in terms of the number of function evaluations that it requires. Powell's method (§10.5) is almost surely faster in all likely applications. However the downhill simplex method may frequently be the best method to use if the figure of merit is "get something working quickly" for a problem whose computational burden is small.

The method has a geometrical naturalness about it which makes it delightful to describe or work through:
A simplex is the geometrical figure consisting, in N dimensions, of N + 1 points (or vertices) and all their interconnecting line segments, polygonal faces, etc. In two dimensions, a simplex is a triangle. In three dimensions it is a tetrahedron, not necessarily the regular tetrahedron. (The simplex method of linear programming also makes use of the geometrical concept of a simplex. Otherwise it is completely unrelated to the algorithm that we are describing in this section.) In general we are only interested in simplexes that are non-degenerate, i.e.. which enclose a finite inner N~dimensional volume. If any point of a non-degenerate simplex is taken as the origin, then the N other points define vector directions that span the N-dimensional vector space.
In one-dimensional minimization, it was possible to bracket a minimum so that the success of a subsequent isolation was guaranteed. Alas! There is no analogous procedure in multidimensional space. For multidimensional minimization, the best we can do is give our algorithm a starting guess, that is, an N~vector of independent variables as the first point to try. The algorithm is then supposed to make its own way downhill through the unimaginable complexity of an N-dimensional topography, until it encounters an (at 1least local) minimum.
The downhill simplex method must be started not just with a single point, but with N + 1 points, defining an initial simplex. If you think of one of these points (it matters not which) as being your initial starting point P0, then you can take the other N points to be

$$P_i = P_0 + \lambda e_i \qquad (10.4.1)$$

where the ei's are N unit vectors, and where λ is a constant which is your guess of the problem's characteristic length scale. (Or, you could have different λ_i's for each vector direction).

The downhill simplex method now takes a series of steps, most steps just moving the point of the simplex where the function is largest ("highest point") through the opposite face of the simplex to a lower point. These steps are called reflections, and they are constructed to conserve the volume of the simplex (hence maintain its non-degeneracy). When it can do so, the method expands the simplex in one or another direction to take larger steps. When it reaches a "valley floor," the method contracts itself in the transverse direction and tries to ooze down the valley. If there is a situation where the simplex is trying to "pass through the eye of a needle," it contracts itself in all directions, pulling itself in around its lowest (best) point. The routine name AMOEBA is intended to be descriptive of this kind of behavior; the basic moves are summarized in Figure 10.4.1.

Termination criteria can be delicate in any multidimensional minimization routine. Without bracketing, and with more than one independent variaable, we no longer have the option of requiring a certain tolerance for a single independent variable. We typically can identify one "cycle" or "step" of our multidimensional algorithm. It is then possible to terminate when the vector

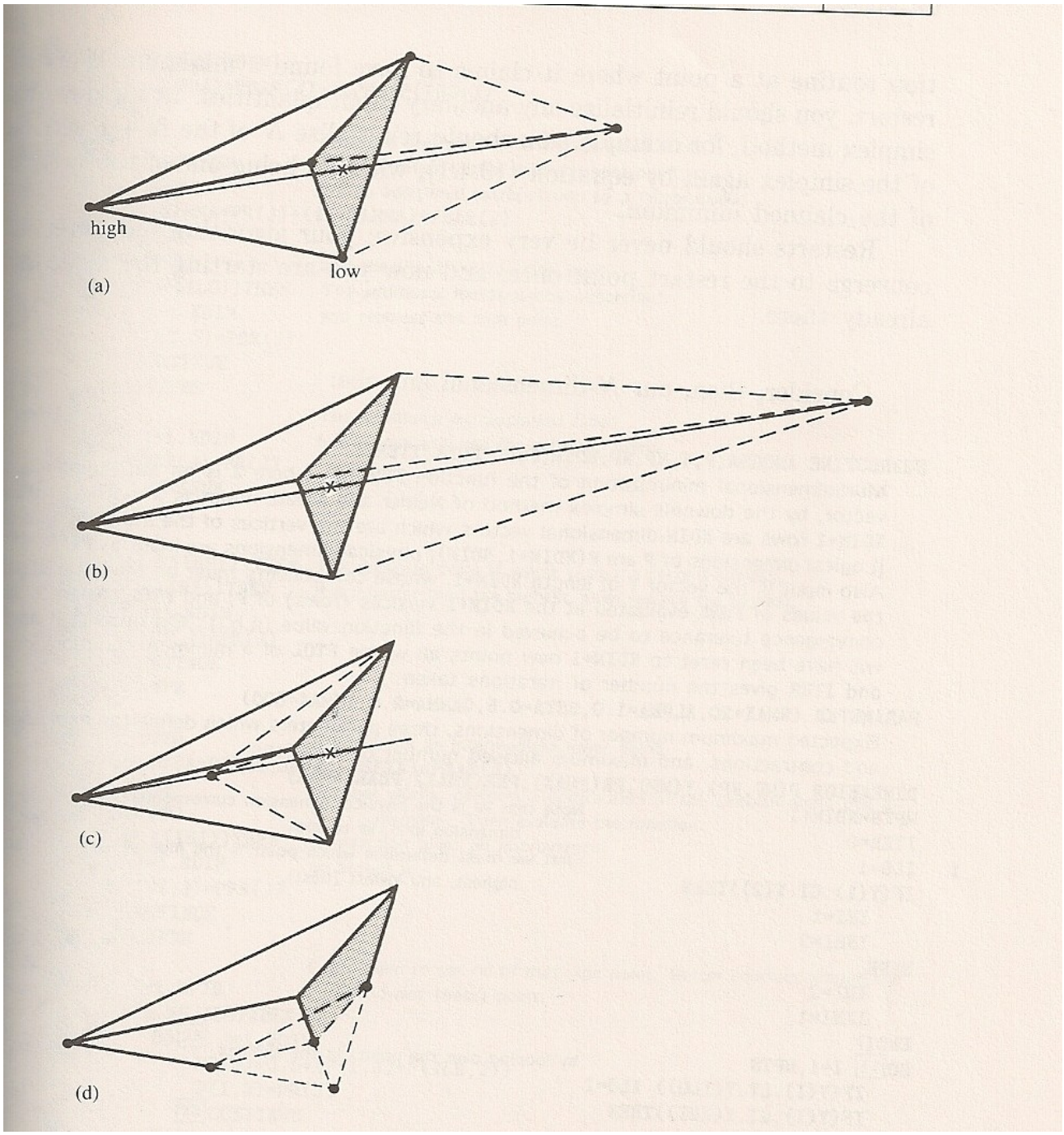

Figure 10.4.1. *Possible outcomes for a step in the downhill simplex method. The simplex at the beginning of the step, here a tetrahedron, is drawn with solid lines. The simplex at the end of the step (drawn dashed) can be either (a) a reflection away from the high point, (b) a reflection and expansion away from the high point, (c) a contraction along one dimension from the high point, or (d) a contraction along all dimensions toward the low point. An appropriate sequence of such steps will always converge to a minimum of the function.*

distance moved in that step is fractionally smaller in magnitude than some tolerance TOL. Alternatively, we could require that the decrease in the function value in the terminating step be fractionally smaller than some tolerance FTOL.

Note that while TOL should not usually be smaller than the square root of the machine precision, it is perfectly appropriate to let FTOL be of order of the machine precision (or perhaps slightly larger so as not to be diddled by round-off).
Note well that either of the above criteria might be fooled by a single anomalous step that, for one reason or another, failed to get anywhere. Therefore, it is frequently a good idea to restart a multidimensional minimization routine at a point where it claims to have found a minimum. For this restart, you should reinitialize any ancillary input quantities. in the downhill simplex method, for example, you should reinitialize N of the N + 1 vertices of the simplex again by equation (10.4.1), with Po being one of the vertices of the claimed minimum.

Restarts should never be very expensive; your algorithm did, after all, converge to the restart point once, and now you are starting the algorithm already there.

[From BIBLI 08].