1. NONLINEAR VECTOR OPERATOR EQUATIONS

1.1. Multivariate root-finding algorithms

Consider now nonlinear finite-dimensional mappings $\boldsymbol{f}: \mathbb{R}^d \to \mathbb{R}^d$, and the root-finding problem

$$\boldsymbol{f}(\boldsymbol{x}) = \mathbf{0}, \tag{1}$$

whose set of solutions generalize the linear mapping concept of a null space, $N(\boldsymbol{A}) = \{\boldsymbol{x} | \boldsymbol{A}\boldsymbol{x} = \mathbf{0}, \boldsymbol{A} \in \mathbb{C}^{d\times d}\}$. As in the scalar-valued case, algorithms are sought to construct an approximating sequence $\{\boldsymbol{x}_k\}_{k\in\mathbb{N}}$ whose limit is a root of (1), by approximating $\boldsymbol{f}$ with $\boldsymbol{g}_k$, and solving

$$\boldsymbol{g}_k(\boldsymbol{x}) = 0. \tag{2}$$

Multivariate approximation is however considerably more complex than univariate approximation. For example, consider $d = 2, \boldsymbol{f}: \mathbb{R}^2 \to \mathbb{R}^2$, and the univariate monomial interpolants in Lagrange form

$$\mathscr{L}_t \boldsymbol{f}(s,t) = \sum_{i=0}^{m} \boldsymbol{f}(x_i, t)\, l_i^x(s),\, \mathscr{L}_s \boldsymbol{f}(s,t) = \sum_{j=0}^{n} \boldsymbol{f}(s, y_j)\, l_j^y(t),$$

with

$$l_i^x(s) = \prod_{k=0}^{m}{}' \frac{s - x_k}{x_i - x_k}, l_j^y(s) = \prod_{l=0}^{n}{}' \frac{t - y_l}{y_j - y_l}.$$

The operator $\mathscr{L}_t$ carries out interpolation at fixed t value of the data set $\mathscr{D}_x = \{(x_i, \boldsymbol{f}(x_i, t)), i = 0, \ldots, m\}$. Similarly, operator $\mathscr{L}_s$ carries out interpolation at fixed s value of the data set $\mathscr{D}_y = \{(y_j, \boldsymbol{f}(s, y_j)), j = 0, \ldots, n\}$. Multivariate interpolation of the data set

$$\mathscr{D} = \{(x_i, y_j, \boldsymbol{f}(x_i, y_j)), i = 0, \ldots, m, j = 0, \ldots, n\},$$

can be carried out through multiple operator composition procedures.

Operator product. Define $\mathscr{L} = \mathscr{L}_t \otimes \mathscr{L}_s$ as

$$\mathscr{L}\boldsymbol{f}(s,t) = (\mathscr{L}_t \mathscr{L}_s)\boldsymbol{f}(s,t) = \mathscr{L}_t(\mathscr{L}_s \boldsymbol{f}(s,t)) = \mathscr{L}_t\left(\sum_{i=0}^{m} \boldsymbol{f}(x_i,t)\, l_i^x(s)\right) = \sum_{i=0}^{m}\sum_{j=0}^{n} \boldsymbol{f}(x_i, y_j)\, l_i^x(s)\, l_j^y(t).$$

Operator Boolean sum. Define $\mathscr{L} = \mathscr{L}_t \oplus \mathscr{L}_s$ as $\mathscr{L} = \mathscr{L}_t + \mathscr{L}_s - \mathscr{L}_t \mathscr{L}_s$

$$\mathscr{L}\boldsymbol{f}(s,t) = \sum_{i=0}^{m} \boldsymbol{f}(x_i,t)\, l_i^x(s) + \sum_{j=0}^{n} \boldsymbol{f}(s,y_j)\, l_j^y(t) - \sum_{i=0}^{m}\sum_{j=0}^{n} \boldsymbol{f}(x_i,y_j)\, l_i^x(s)\, l_j^y(t).$$

1.1.1. First-degree polynomial approximants

Secant method. Bivariate ($d = 2$) root-finding algorithms already exemplifies the additional complexity in constructing root finding algorithms. The goal is to determine a new approximation (x_k, y_k) from the prior approximants

$$(x_0, y_0), \ldots, (x_{k-2}, y_{k-2}), (x_{k-1}, y_{k-1}).$$

Whereas in the scalar case two prior points allowed construction of a linear approximant, the two points in data

$$\mathscr{D} = \{(x_{k-2}, y_{k-2}), (x_{k-1}, y_{k-1})\}$$

are insufficient to determine

$$\mathscr{L}\boldsymbol{f} = \sum_{i=k-2}^{k-1}\sum_{j=k-2}^{k-1} \boldsymbol{f}(x_i, y_j)\, l_i^x(s)\, l_j^y(t),$$

which requires four data points. Various approaches to exploit the additional degrees of freedom are available, of which the class of quasi-Newton methods finds widespread applicability.

Newton, quasi-Newton methods. A linear multivariate approximant in d dimensions requires 2^d data. A Hermite interpolant based upon function and partial derivative values can be constructed, but it is more direct to truncate the multivariate Taylor series

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x}_k) + \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}(\boldsymbol{x}_k)(\boldsymbol{x}-\boldsymbol{x}_k) + \cdots,$$

where

$$\boldsymbol{J} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_d} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_d} \\ \vdots & \vdots & \ddots & \\ \frac{\partial f_d}{\partial x_1} & \frac{\partial f_d}{\partial x_2} & \cdots & \frac{\partial f_d}{\partial x_d} \end{bmatrix} = \nabla \boldsymbol{f},$$

is the Jacobian matrix of $\boldsymbol{f}$. Setting $\boldsymbol{f}(\boldsymbol{x}_{k+1}) = \boldsymbol{0}$, as the condition for the next iterate leads to the update

$$\boldsymbol{J}(\boldsymbol{x}_k)(\boldsymbol{x}_{k+1}-\boldsymbol{x}_k) = -\boldsymbol{f}(\boldsymbol{x}_k),$$

a linear system that is solved at each iteration. Computation of the multiple partial derivatives arising in the Jacobian might not be possible or too expensive, hence approximations are sought $\boldsymbol{B}_k \cong \boldsymbol{J}(\boldsymbol{x}_k)$, similar in principle to the approximation of a tangent by a secant. In such quasi-Newton methods, a secant condition on $\boldsymbol{B}_k$ is stated as

$$\boldsymbol{B}_k(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) = \boldsymbol{f}(\boldsymbol{x}_k) - \boldsymbol{f}(\boldsymbol{x}_{k-1}),$$

and corresponds to a truncation of the Taylor series expansion around $\boldsymbol{x}_{k-1}$. The above secant condition is not sufficient by itself to determine $\boldsymbol{B}_k$, hence additional considerations can be imposed.

1. Recalling that the scalar Newton method for finding roots of $f(x) = 0$ converges in a region where $f', f'' > 0$, imposing analogous behavior for $\boldsymbol{B}_k$ suggests itself. This is typically done by requiring $\boldsymbol{B}_k$ to be symmetric positive definite.
2. Assuming convergence of the approximating sequence $\{\boldsymbol{x}_k\}_{k\in\mathbb{N}}$ to a root, $\boldsymbol{B}_{k+1}$ should be close to the previous approximation suggesting the condtion

$$\min_{\boldsymbol{B}_{k+1}} \|\boldsymbol{B}_{k+1} - \boldsymbol{B}_k\|.$$

Various algorithms arise from a particular choice of norm and procedure to apply (2).

One widely used quasi-Newton method, arising from a rank-two update at each iteration to maintain positive definiteness, is the Broyden-Fletcher-Goldfard-Shanno update

$$\boldsymbol{B}_{k+1} = \boldsymbol{B}_k + \frac{\boldsymbol{y}_k \boldsymbol{y}_k^T}{\boldsymbol{y}_k^T \boldsymbol{s}_k} - \frac{\boldsymbol{B}_k \boldsymbol{s}_k \boldsymbol{s}_k^T \boldsymbol{B}_k^T}{\boldsymbol{s}_k^T \boldsymbol{B}_k \boldsymbol{s}_k},$$

where the updates are determined by

1. Solving $\boldsymbol{B}_k \boldsymbol{p}_k = -[\boldsymbol{f}(\boldsymbol{x}_k) - \boldsymbol{f}(\boldsymbol{x}_{k-1})]$ to find a search direction $\boldsymbol{p}_k$;
2. Finding the distance along the search direction by $\alpha_k = \operatorname{argmin} \|\boldsymbol{f}(\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k)\|_2$;
3. Updating the approximation $\boldsymbol{s}_k = \alpha_k \boldsymbol{p}_k$, $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{s}_k$
4. Computing $\boldsymbol{y}_k = \boldsymbol{f}(\boldsymbol{x}_{k+1}) - \boldsymbol{f}(\boldsymbol{x}_k)$.