On a modern interpretation of sequential approximate optimization

Dirk Munro (Hamburg, Germany)

February 28, 2022

*This note is intended as a general yet simple introduction to the subject. No novelty beyond the interpretation, description (and notation) is introduced. The list of citations would not be reasonably complete without those referenced in the **Background**, in closure of the note.*

1 Introduction

Assume we have an optimization problem $\mathcal{P}$ and an array of scalar decision variables $\mathbf{x}$. Each decision variable is assumed to be continuous, and collected in an array of length n; *i.e.* $\mathbf{x} \in \mathbb{R}^n$. What is meant (herein[*]) by 'an optimization problem $\mathcal{P}$'? We will attempt a modern description, from a solution-method point-of-view. We assume that $\mathcal{P}$ is programmatic (numeric) entity, which may be evaluated with an array of particular decision (input) variable values $\mathbf{x}$, $\mathcal{P}[\mathbf{x}]$[†], and, in so doing, the problem returns the following information

$$\left\{ \begin{array}{r} \mathbf{c} = (\mathrm{c}_0, \mathrm{c}_1, \dots, \mathrm{c}_m) \\ \partial\mathbf{c} = (\partial\mathbf{c}_0, \partial\mathbf{c}_1, \dots, \partial\mathbf{c}_m) \\ \underline{\overline{\mathbf{x}}} = (\underline{\mathbf{x}},\, \overline{\mathbf{x}}) \end{array} \right\} = \mathcal{P}[\mathbf{x}]\,. \tag{1}$$

That is

(i) an array of scalar-valued cost-and-constraint functions $\mathbf{c}$, of length m. We will adopt the convention that the first entry c_0 is a scalar-valued objective (cost) to be minimised, and the following $m-1$ entries are scalar-valued constraint function values, with values of $\mathrm{c}_j \leq 0$ denoting feasibility; *i.e.* adherence to constraint j, and violation otherwise;

(ii) assuming each cost and constraint function is (at least once) continuously differentiable, the first-order derivatives of each, to each decision variable, $\partial\mathbf{c}$, with

$$\partial\mathbf{c}_j = \left(\frac{\partial \mathrm{c}_j}{\partial \mathrm{x}_1}, \frac{\partial \mathrm{c}_j}{\partial \mathrm{x}_2}, \dots, \frac{\partial \mathrm{c}_j}{\partial \mathrm{x}_n} \right) \quad \text{for} \quad j = 0, 1, \dots, m\,; \tag{2}$$

(iii) the lower $\underline{\mathbf{x}}$ and upper $\overline{\mathbf{x}}$ bounds[‡] of the decision space, respectively, denoted by $\underline{\overline{\mathbf{x}}}$.

[*]It is open to (largely academic) debate whether or not the definition implied here-in is traditionally correct. Traditionally, the 'output' of the optimization problem $\mathcal{P}$ may be seen to be the decision variables at solution of the problem $\mathbf{x}^*$. Herein we follow a more practical interpretation, insofar as we say that the problem $\mathcal{P}$ itself does not provide the solution $\mathbf{x}^*$ as 'output' (which is true).

[†]Through-out square brackets $[\cdot]$ will denote an operation of 'to evaluate at'; and it may be used and dropped freely as is relevant to the description of a particular entity, in a particular context.

[‡]It is again interesting (but somewhat arbitrary) whether or not the (global) bounds on the decision space $\underline{\overline{\mathbf{x}}}$ is taken to be defined in, and provided by, the problem $\mathcal{P}$. Practically, it is perhaps useful to interpret the bounds provided by problem $\mathcal{P}$ as restrictions on the extent of the decision space $\underline{\overline{\mathbf{x}}}$ which follow naturally from the physics or character of the decision variables—*e.g.* the range of thickness (greater than zero, but not infinite) permitted to a structural member. But of course, the decision / optimization algorithm, which operates on the information provided by problem $\mathcal{P}$, may (and typically will) operate iteratively in smaller decision spaces, overwriting the bounds provided by problem $\mathcal{P}$.

In other words, beyond the information we can collect from the quantities returned for particular evaluations $\{\mathbf{c}, \partial\mathbf{c}, \underline{\overline{\mathbf{x}}}\} = \mathcal{P}[\mathbf{x}]$ we have no further information of the problem $\mathcal{P}$, except for what can be deduced from the upstream assumptions (and knowledge of the nature of the problem). Moreover, in general, a single evaluation of problem $\mathcal{P}[\mathbf{x}]$ is considered computationally expensive—or inconvenient, at least—insofar as it is non-trivially costly to evaluate in terms of computational resources and/or available channels of computational integration. That is simulation-based, with multiple, potentially large-scale finite-element analyses, but also computationally cheap (*e.g.* reduced-order) analyses, which sit behind licensed applications and/or require significant network communications (*e.g.* cloud-based microservices).

How to find a candidate[§] optimum solution $\mathbf{x}^*$ of problem $\mathcal{P}$? Well, we can start by sampling the decision space $\underline{\overline{\mathbf{x}}}$, to find an $\mathbf{x}$ with a reasonable cost $\mathrm{c}_0[\mathbf{x}]$, while all constraints are feasible $\mathrm{c}_j[\mathbf{x}] \leq 0$, $\forall j > 0$. Then, we can ask the question: can this sample decision $\mathbf{x}$ be improved if we adjust the evaluation in the decision space by a particular amount $\mathbf{x} + \Delta\mathbf{x}$? If there is a $\Delta\mathbf{x}$ for which $\mathrm{c}_0[\mathbf{x}] > \mathrm{c}_0[\mathbf{x} + \Delta\mathbf{x}]$, while all constraints remain feasible $\mathrm{c}_j[\mathbf{x} + \Delta\mathbf{x}] \leq 0$, $\forall j > 0$, then the sample $\mathbf{x}$ we started with was indeed not optimal, and we should adjust the decision to $\mathbf{x} + \Delta\mathbf{x}$.

How to determine what $\Delta\mathbf{x}$ should be? Keep in mind, we have assumed we are working with an expensive-to-evaluate problem $\mathcal{P}$, which only returns the information $\{\mathbf{c}, \partial\mathbf{c}, \underline{\overline{\mathbf{x}}}\} = \mathcal{P}[\mathbf{x}]$ upon evaluation at $\mathbf{x}$. In practical terms, we will take this to mean that we want to compute a good change in the decision space $\Delta\mathbf{x}$, without re-evaluating problem $\mathcal{P}$.

Let us, in hope[¶], assume that the actual cost-and-constraint functions $\mathbf{c}$ are (near) linear over the entire decision space $\underline{\overline{\mathbf{x}}}$. That is, an analytic linear approximation of the cost-and-constraint functions $\mathbf{c}$, in terms of $\Delta\mathbf{x}$, is constructed

$$\mathbf{v}[\Delta\mathbf{x}] = \mathbf{c} + \partial\mathbf{c} \cdot \Delta\mathbf{x}\,, \tag{3}$$

with first derivatives following accordingly

$$\partial\mathbf{v}[\Delta\mathbf{x}] = \partial\mathbf{c}\,. \tag{4}$$

In general, the first-order approximation of the cost-and-constraint functions $\mathbf{v}$ is only equal[‖] to the actual cost-and-constraint functions $\mathbf{c}$ at $\mathbf{x}$—i.e., with $\Delta\mathbf{x} = \mathbf{0}$—and representative in a infinitely small region around $\mathbf{x}$. In the light of this, we introduce a lower $\Delta\underline{\mathbf{x}}$ and upper $\Delta\overline{\mathbf{x}}$ bound on the change $\Delta\mathbf{x}$ we allow ourselves in the decision space $\underline{\overline{\mathbf{x}}}$. That is, we have arrived at a subproblem

$$\left\{\begin{aligned} \mathbf{v} &= (\mathrm{v}_0, \mathrm{v}_1, \ldots, \mathrm{v}_m) \\ \partial\mathbf{v} &= (\partial\mathbf{v}_0, \partial\mathbf{v}_1, \ldots, \partial\mathbf{v}_m) \\ \Delta\underline{\overline{\mathbf{x}}} &= (\Delta\underline{\mathbf{x}},\ \Delta\overline{\mathbf{x}}) \end{aligned}\right\} = \mathcal{S}[\Delta\mathbf{x}]\,, \tag{5}$$

which is, importantly, unlike problem $\mathcal{P}$, very cheap to evaluate for different values of $\Delta\mathbf{x}$. In general we say that a solution to problem $\mathcal{S}$—finding that $\Delta\mathbf{x}$ in $\Delta\underline{\overline{\mathbf{x}}}$ for which v_0 is a minimum, while all $\mathrm{v}_j \leq 0$, $\forall j > 0$—is computable in polynomial time. Moreover, in practice, polynomial time solution methods are readily available.

[§] In general the optimization problem is assumed to be multi-modal. The topic is discussed, and a solution method is proposed, in [1].

[¶] or desperation.

[‖] Equal in zero– and first-order information.

We may thus, upon solving subproblem $\mathcal{S}$, update the decision to $\mathbf{x} \leftarrow \mathbf{x} + \Delta\mathbf{x}$; repeat the evaluation of problem $\mathcal{P}$ at the new values of the decision variables $\{\mathbf{c}, \partial\mathbf{c}, \overline{\underline{\mathbf{x}}}\} = \mathcal{P}[\mathbf{x}]$, and repeat the construction and solution of subproblem $\mathcal{S}$ *ad infinitum*, or, until there is no change in the decision $\Delta\mathbf{x}$ which improves the solution $\mathbf{x} \rightarrow \mathbf{x}^*$ of problem $\mathcal{P}$, further.

In general, however, linear approximations $\mathbf{v}$ of the cost-and-constraint functions $\mathbf{c}$ may result in changes $\Delta\mathbf{x}$ in the decision space which violates the constraints $\mathrm{c}_j[\mathbf{x} + \Delta\mathbf{x}] > 0\ \forall j > 0$, increases in the cost function $\mathrm{c}_0[\mathbf{x}+\Delta\mathbf{x}]$, very restrictive allowable decision changes $\Delta\overline{\underline{\mathbf{x}}}$ resulting in excessive, expensive evaluations of problem $\mathcal{P}[\mathbf{x}]$, and/or complete failure of the procedure to converge to a reasonable solution $\mathbf{x}^*$ of problem $\mathcal{P}$.

What can we do?

2 Decision space transformations

Let us assume that we notice, by observation, that the cost-and-constraint functions $\mathbf{c}$ have a particular proportional relationship to decisions $\mathbf{x}$ in the decision space $\overline{\underline{\mathbf{x}}}$. For example, we might notice[**] that the cost-and-constraint functions $\mathbf{c}$ have a reciprocal-like relation to our decision variables, *i.e.* $\mathbf{c} \sim 1/\mathbf{x}$. Can we exploit this to improve the change $\Delta\mathbf{x}$ in the decision space we attempt to make? Yes, we can imagine applying an analytic transformation (mapping) to the decision space $\mathbf{y} = \mathbf{y}[\mathbf{x}]$, and formulating the approximate cost-and-constraint functions $\mathbf{v}$ in terms of it

$$\mathbf{v} = \mathbf{c} + \partial_{\mathbf{y}}\mathbf{c} \cdot \Delta\mathbf{y} \,. \tag{6}$$

That is, we hope that the transformation of the decions space from $\mathbf{x} \rightarrow \mathbf{y}$ has worked to 'linearise' the cost-and-constraint functions $\mathbf{c}$. Rewriting Eq. (6) in terms of the first-order information supplied by problem $\mathcal{P}$, we retrieve

$$\mathbf{v} = \mathbf{c} + \partial\mathbf{c} \cdot \partial_{\mathbf{x}}^{-1}\mathbf{y} \cdot \Delta\mathbf{y} \,, \tag{7}$$

with the derivative of the analytic mapping $\mathbf{y} = \mathbf{y}[\mathbf{x}]$ easy to compute

$$\partial_{\mathbf{x}}^{-1}\mathbf{y} = 1 \Big/ \frac{\partial\mathbf{y}}{\partial\mathbf{x}} \,. \tag{8}$$

Note that $\partial_{\mathbf{x}}^{-1}\mathbf{y}$ is an $n \times n$ square matrix (although in practice, typically, only diagonal terms are utilised).

How does this help us? Consider again the example of the reciprocal transformation $\mathbf{y} = 1/\mathbf{x}$. In this case

$$\partial_{\mathbf{x}}^{-1}\mathbf{y} = -\mathbf{x}_{\mathbf{I}}^2 \,, \tag{9}$$

with $_{\mathbf{I}}$ indicating that the array is cast along the diagonal of the correspondingly sized square identity matrix. Hence

$$\mathbf{v} = \mathbf{c} + \partial\mathbf{c} \cdot (-\mathbf{x}_{\mathbf{I}}^2) \cdot \Delta\mathbf{y} \,. \tag{10}$$

The approximate cost-and-constraint functions $\mathbf{v}$ may be re-written in terms of the original decision space $\mathbf{x}$ and the change $\Delta\mathbf{x}$, which yields

$$\mathbf{v} = \mathbf{c} + \partial\mathbf{c} \cdot (-\mathbf{x}_{\mathbf{I}}^2) \cdot \left(\frac{1}{\mathbf{x} + \Delta\mathbf{x}} - \frac{1}{\mathbf{x}} \right) \,, \tag{11}$$

[**]and/or know, based on knowledge on the nature of the problem $\mathcal{P}$.

and, is the same as

$$\mathbf{v} = \mathbf{c} + \partial\mathbf{c} \cdot \left(\frac{\mathbf{x}}{\mathbf{x}+\Delta\mathbf{x}}\right)_{\mathbf{I}} \cdot \Delta\mathbf{x}\,. \tag{12}$$

Notice how the term in brackets $(\cdots)$ has introduced a analytically available nonlinearity (or 'curvature') in the approximate cost-and-constraint functions $\mathbf{v}$, with respect to $\Delta\mathbf{x}$. Keep in mind, the evaluation of problem $\mathcal{P}[\mathbf{x}]$ is constant while we repeatedly evaluate (and solve) subproblem $\mathcal{S}[\Delta\mathbf{x}]$—which remains cheap to evaluate—to compute the change in the decision space $\Delta\mathbf{x}$.

Similarly, a decision space transformation may be generalised to the form $\mathbf{y} = \mathbf{x}_{\mathbf{I}}^{a}$, with a any real number. In this case

$$\partial_{\mathbf{x}}^{-1}\mathbf{y} = \frac{1}{a}\mathbf{x}_{\mathbf{I}}^{1-a}\,, \tag{13}$$

and hence

$$\mathbf{v}[\Delta\mathbf{x}] = \mathbf{c} + \partial\mathbf{c} \cdot \frac{1}{a}\mathbf{x}_{\mathbf{I}}^{1-a} \cdot \left((\mathbf{x}+\Delta\mathbf{x})^{a} - \mathbf{x}^{a}\right)\,, \tag{14}$$

which is easily evaluated for different values of $\Delta\mathbf{x}$, with any manner of estimate for a. The first derivates of the approximate cost-and-constraint functions $\mathbf{v}$ follow readily

$$\partial\mathbf{v}[\Delta\mathbf{x}] = \partial\mathbf{c} \cdot (\mathbf{x}^{1-a}) \cdot (\mathbf{x}+\Delta\mathbf{x})^{a-1}\,, \tag{15}$$

with the first-order information of problem $\mathcal{P}$ at the current evaluation point $\mathbf{x}$, $\partial\mathbf{c}$, maintained

$$\partial\mathbf{v}[\mathbf{0}] = \partial\mathbf{c}\,. \tag{16}$$

3 Taylor series expansions

The analytic nonlinearity introduced by the decision space transformation is conveniently unpacked by Taylor series expansion $\boldsymbol{\nu}$, of the transformed cost-and-constraints $\mathbf{v}$, in terms of $\Delta\mathbf{x}$, away from the current decision point $\mathbf{x}$ (where $\Delta\mathbf{x} = 0$)

$$\boldsymbol{\nu}[\Delta\mathbf{x}] = \mathbf{v}[\mathbf{0}] + \frac{1}{1!}\partial\mathbf{v}[\mathbf{0}] \cdot \Delta\mathbf{x} + \frac{1}{2!}\partial^2\mathbf{v}[\mathbf{0}] \cdot \Delta\mathbf{x}^2 + \frac{1}{3!}\partial^3\mathbf{v}[\mathbf{0}] \cdot \Delta\mathbf{x}^3 + \ldots. \tag{17}$$

For sake of simplicity, let us truncate after the first nonlinear (2nd) term

$$\boldsymbol{\nu}[\Delta\mathbf{x}] = \mathbf{v}[\mathbf{0}] + \partial\mathbf{v}[\mathbf{0}] \cdot \Delta\mathbf{x} + \frac{1}{2}\partial^2\mathbf{v}[\mathbf{0}] \cdot \Delta\mathbf{x}^2\,, \tag{18}$$

noting that the second-derivative of the transformed cost-and-constraints $\mathbf{v}$ is readily available

$$\partial^2\mathbf{v}[\Delta\mathbf{x}] = \partial\mathbf{c} \cdot (a-1)\mathbf{x}^{1-a} \cdot (\mathbf{x}+\Delta\mathbf{x})^{a-2}\,, \tag{19}$$

and, at the current decision point $\Delta\mathbf{x} = \mathbf{0}$, it reduces to

$$\partial^2\mathbf{v}[\mathbf{0}] = \partial\mathbf{c} \cdot \left(\frac{a-1}{\mathbf{x}}\right)_{\mathbf{I}}\,. \tag{20}$$

This notion of 'approximated approximations' was first introduced, in this manner, by Groenwold *et al.* [2]. In significant practical terms, the 2nd-order Taylor expansion allows for a completely flexible and general interface with common polynomial time (subproblem) solvers, which are typically fed (as cheaply as possible) by the first and second-order information of the subproblem at different values of $\Delta\mathbf{x}$[††].

Comment on introduction of curvature information... and that pretty much anything can be done... also analytic, if available... QPQP .... cone programs... dual methods ...

4 Quadratic programs and the necesary conditions

Solution methods for the form

in this way, the notion of decicsion space transformations become obsolete to some extent, in practical terms. Althopugh the notion of it may be used to estimate the second order information...

in rough drafting If we use only diagonal information. Ensure it is positive definite. and formulate the hessian of the lagrangian ...

Then we arrive at a convenient computational 'kernel': the quadratic program (QP)[‡‡]. In simple terms, polynomial time solution methods for QPs are often the cheapest, fastest, simplest to implement, interface, and most readily available (to an average computational engineer), and ... for subproblem $\mathcal{S}$, given some straight-forward conditions. See for example the recently released implementation by Boyd and collaborators [3].

TO BE COMPLETED

Eventually we arrive rather naturally at the necessary conditions: How far can we go, before we have to evaluate the problem again..? Keeping in mind, that we can not afford to evaluate the problem an excessive number of times... etc. etc. Move limits and estimates of nonlinear information (surrogate functions/problems, second order estimates).... upon convergence / when we can stop, we get something for free: the necessary conditions are satisfied / we have arrived at the necessary conditions in a rather practical way. Done.

Background

The introduction to the 'Elements of Structural Optimization' by Haftka, Gürdal and Kamat [4]. The review of sensitivity analysis in structural optimization by van Keulen, Haftka and Kim [5]. More to follow.

References

[1] An interpretation of Bayesian global optimization as proposed by Snyman and Fatti. `https://github.com/dirkmunro89/SOAPs/blob/main/bayopt/bay_20220104.pdf`. Accessed: 2022-02-27.

[2] A.A. Groenwold, L.F.P. Etman, and D.W. Wood. Approximated approximations for SAO. *Structural and Multidisciplinary Optimization*, 41(1):39–56, 2010.

[††]Even if linear programming is applied to solve the subproblem $\mathcal{S}$, the (variable) first-order information $\partial\boldsymbol{\nu}[\Delta\mathbf{x}] = \partial\mathbf{v}[\mathbf{0}] + \partial^2\mathbf{v}[\mathbf{0}]\cdot(\mathbf{x}+\Delta\mathbf{x})$ will be supplied, iteratively.

[‡‡]Conceptually, the step-wise solving of QP problems in general, non-linear programming, can be seen as equivalent to the way in which a linear system-solve is the 'kernel' in nonlinear structural analysis. That is the 'kernel' of the Newton-Rhapson method.

[3] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020.

[4] R.T. Haftka, Z. Gürdal, and M.P. Kamat. *Elements of structural optimization.* Kluwer Academic Publishers, Dordrecht, The Netherlands, 1990.

[5] F. Van Keulen, R.T. Haftka, and N.H. Kim. Review of options for structural design sensitivity analysis. Part 1: Linear systems. *Computer methods in applied mechanics and engineering*, 194(30):3213–3243, 2005.