

# A review of constraint management

H.P. Wynn  
London School of Economics, UK

## 1 Background

There are several areas of MUCM in which constraint management may have a role and this is the reason it was included as a task. This review is simply for discussion rather than making choices about what methods may be used.

At the most basic level input and outputs spaces are typically constraint regions in say  $\mathcal{X} \subset R^p$  and  $\mathcal{Y} \subset R^q$  respectively. The simulator is a function

$$y = f(x) : \mathcal{X} \rightarrow \mathcal{Y},$$

and the emulator is another such function

$$\hat{y} = \hat{f}(x) : \mathcal{X} \rightarrow \mathcal{Y},$$

In standard cases we may consider that  $\mathcal{X}$  is a hyper-rectangle:

$$\mathcal{H} = \{x : a_i \leq x_i \leq b_i, i = 1, \dots, p\},$$

and similarly for  $\mathcal{Y}$ . We may also standardize the region with a shift and scale transformation, for example so that  $(a_i, b_i) = (0, 1)$ ,  $i = 0, \dots, p$ . The experimental design typically, but not always, occupies the constrained region,  $\mathcal{H}$ . In fact sometimes it is convenient to take a discrete *candidate set* sat  $D_N \subset \mathcal{H}$ , of total size  $N$  and consider the actual design as a subset  $D_n \subset D_N \subset \mathcal{X}$ , of *sample size*  $n$ .

For the moment we concentrate on input regions. In many practical situations the actual input region (i) may be more complex region than a hyper-rectangle (ii) may have much smaller volume than, for example, the small rectangle containing it. Several problems may arise:

1. describing the region in a parsimonious way
2. updating the region if new information arises
3. constructing candidate sets and experimental designs for constrained regions

We review the considerable range of other areas where such problems have arisen so that we may draw on appropriate methods for the MUCM project.

## 2 Computational geometry

Perhaps the best overall term for the kinds of mathematical and computational problems which arise in constraint management is *computational geometry*. Sometimes when there are polynomials involved we may talk about computational algebraic geometry. For example, a disc is a circular area

$$(x_1 - a)^2 + (x_2 - b)^2 \leq c$$

and incorporates a constraint and therefore is typical of the kind of objects dealt with in computational geometry. But it also comprises a circle:

$$(x_1 - a)^2 + (x_2 - b)^2 = c,$$

which, being the solution of an algebraic equation, is a variety. A union of algebraic varieties is an *algebraic set*.

Computational geometry probably arose first in *CAD: Computer Aided Design*. This is divided into 2- and 3-dimensional methods, used for represented shapes in engineering, architecture and similar field. Advanced methods also have intelligence, over and above geometric intelligence, where the methods are attached to solvers to computer physical quantities such as stress. In engineering this can be called CAE: computer-aided engineering. Another example of an area which combines advanced geometric representation, graphics and analysis is computational chemistry.

An important idea from these areas is that of a *feature* or a *primitive*. This might be line, arc, surface, rectangle, circle, box, sphere, cone, wedge etc. In such cases computational geometry can be thought of as methods for manipulating primitives. In general it is practically easier to code in terms of a symbolic script language for the primitives. The finished product, is often called a “model”.

## 3 Boolean operations

The algebra of sets is Boolean algebra and Boolean algebra methods are very important in manipulating sets. Let us start with a collection of subsets  $\mathcal{C} = \{C_i\}_{i=1}^N$  of a base set  $\mathcal{X}$ . Assume that the collections is closed under unions and intersections. It then forms a Boolean *lattice* of sets although we shall not dwell on the lattice properties. Let us also include complements:  $\{C_i^c\}$ . Then we obtain  $2^n$  mutually disjoint elementary sets of the form:

$$E_J = \bigcap_{i \in J} C_i \bigcap_{j \in J^c} C_j^c,$$

where  $J \subset \{1, \dots, n\}$  is an index set and  $J^c = \{1, \dots, n\} \setminus J$ .

We can create new sets by taking unions of elementary sets. Since there are, in general,  $2^N$  elementary sets we can obtain up to  $(2^N)!$  sets in this way. We can turn this statement around and say that any set which can be built from the original  $C_i$  is the union of intersection. This is an example of a *disjunctive normal form*. A *conjunctive normal form* is when a set is expressed as an intersection of unions. We can clearly get from one to the other by taking complements and using de Morgan's rules:

$$(C_1 \cup C_2)^c = C_1^c \cap C_2^c$$

The indicator function of a set  $C$  is a function on the base set  $\mathcal{X}$  defined as:

$$I_C(x) = \begin{cases} 1, & \text{if } x \in C \\ 0, & \text{otherwise} \end{cases}$$

Then all the operations of intersection, union and complements can be represented as algebraic operations on indicators:

$$I_{C_1 \cap C_2} = I_{C_1} I_{C_2} \tag{1}$$

$$I_{C_1 \cup C_2} = I_{C_1} + I_{C_2} - I_{C_1} I_{C_2} \tag{2}$$

For an elementary set we have

$$I_{E_J}(x) = \prod_{i \in J} I_i(x) \prod_{j \in J^c} (1 - I_j(x)),$$

where  $I_i(x) = I_{C_i}(x)$ .

An important formula is the inclusion-exclusion identity (IEI), which is the generalisation of (2) the form

$$I_{\cup_i C_i} = \sum_i I_{C_i} - \sum_{i < j} I_{C_i \cap C_j} + \dots (-1)^{n-1} I_{\cap_i C_i},$$

and all the intersection terms can be expressed as products by inductive use of (1). Note that if we truncate the IEI at successive terms we get upper and lower bounds for  $I_{\cup_i C_i}$ , starting with an upper bound. This is familiar from probability theory: taking expectations with respect to some some distribution on  $\mathcal{X}$  we have upper and lower bounds for  $\text{prob}\{\cup_i C_i\}$  called the generalised Bonferroni-Fréchet bound. The first upper bound is simply referred as the Bonferroni bound.

## 4 Complexity reduction via geometry

A simple exercise can convince us that the geometry of the sets,  $C_i$  reduces the complexity of Boolean operations. Here is a simple example. Draw the

Venn diagram of 3 circles in the two dimensional plain. Then, provided the circles are in general position and counting complements, we obtain  $2^3 = 8$  elementary sets. However if we draw 4 circles we obtain no more than 14 elementary sets, not the  $2^4 = 16$  than we might expect: the geometry has decreased the complexity.

*Vapnik-Cernovenkis (VC) dimension.* This is a kind of dual of the phenomenon just explained. Consider 4 points in 2 dimensions. Label some of them good, (1), and the others bad, (0). Then we can always find a circle which separates the good from the bad. But we can find five points and a labeling when this is not possible. The VC dimension of the circle in 2 dimensions is therefore said to be 4. If we take half spaces of the form

$$H(a, b) = \{x : a^T x \leq b\}, \quad (3)$$

for a vector  $a$  and constant  $b$  then the VC dimension reduces to 3. The fundamental nature of VC dimension comes from the idea of separation or discrimination: how can we discriminate between a labeled set of points using a particular shaped feature or primitive. The more complex the primitives the more we are able to discriminate but then the higher the computation cost and the higher the model complexity. The VC dimension controls the rate of learning in the rather non-parametric environment of machine learning where data is often assumed to come from an arbitrary distribution. Specifically it arises in so-called concentration inequalities which control the error rate. [3]

*Voronoi diagrams, discrete tubes* Another way of investigating the reduction in complexity is to ask whether an IEI can be reduced in complexity and in particular whether we need to use all the more complex intersections. It turns out that for circles and spheres in  $R^d$  this is related to Voronoi diagrams. Here is a simple version of the main result. Take 4 points in two dimensions. Construct the Voronoi diagram. Join every pair of points whose Voronoi cells touch (either in a point or an edge). This gives the Delaunay (dual) complex. Take all vertices, pairs of points (edges) and triples (triangles) which feature in the complex. The result is that the reduced IEI holds for circles of equal radius based only on these singletons, pairs and triples. We do not need to use the four-way intersection. This generalizes to dimension  $p$  and we do not need to use more than “depth”  $p + 1$ . [1],[4],[5],[6]

*Hyperplane arrangements, convex sets.* If we draw a collection of lines in 2 dimensions we have what is called a *hyperplane arrangement*. More generally, a collection half-spaces is defined by a set of  $p-1$  hyperplanes in  $R^p$  and we are interested in the elementary sets which are the cells between the hyperplanes. The number of such cells is again restricted by the geometry. For 3 lines in 2 dimensions we obtain at most 7 cells (not  $2^3 = 8$ ). Among many results

there are formulae for the maximum number of such cells and for fixed  $p$ , the number grows polynomially in the number of hyperplanes. [7]

Somewhat related is the intersection of a set of (half) closed halfspaces:  $\mathcal{H} = \cap_i H_i$ , of the form (3). This, as we have mentioned, is a convex set. Consider its complement:

$$\mathcal{H}^c = (\cap_i H_i)^c = \cup_i H_i^c$$

This is a union of (open) halfspaces. We are again back in the IEI problem. The question is: can we again reduce the complexity? The solution is that we only use the intersections which correspond to the *facets* of  $\mathcal{H}$ , that is the live tangent hyperplanes and their intersection which are in the boundary. When the hyperplanes are in general position we need not use, in the IEI, intersections of more than  $p$  half-spaces. This is because no more than  $p$  hyperplanes can meet on the boundary of the convex set. But when the hyperplanes are not in general position it can be more complex. An example is a pointed cone with many tangent hyperplanes meeting at the point of the cone. [4]

*Minkowski tubes, boxes etc* A Minkowski tube (sausage) in the region formed by extending outwards a given object. A good way to visualizing this, for a convex body, in three dimensions is to roll a small ball around touching the object. The tube is the union of all the positions of the ball. Related is an “ordinary” tube. This is the region closest to a, typically curved, axis. If the axis is not too bent and the ball radius is small enough the volume of the tube is, except for edge effects is the cross sectional area times the length of the axis. This is the Weyl tube theory and is typically taken to be a branch of *integral geometry*. The full “Weyl tube formula” covers the case where the tube may twist into itself. It is intuitively clear that the formula will depend on various “curvature tensors”. The theory has also been use morphology, multiple comparison, up crossing of stochastic processes and other fields. [8]

## 5 Constraints

In areas such as optimisation, constraints are written in the form

$$g_j(x) \leq b_j, \quad j = 1, \dots, k$$

and the type of constraint is specified in terms of the type of function. Thus linear constraints take the form

$$a_j^T x \leq b_j,$$

and we see immediately that each constraint is a half space and the full constraint set is the intersection of half spaces, which is a convex set.

It is convenient to think in terms of level sets and sub-level set functions. A level set is the solution of the equality constraint  $\{x : g(x) = b\}$  and a sub-level set is the solution of the inequality constraint  $\{x : g(x) \leq b\}$ . Thus asking for all the inequality constraints to hold is equivalent to defining an intersection. Then we are essentially in the same situation as in the last section and may ask given a set of constraints, what is the complexity? We can go further and build constraint regions by taking unions of intersections.

*Parameters.* We have seen that constraints are often defined using parameters and we need to be little careful that the parametrisation is unique. A halfspace is unique if we restrict  $a$  in (3) to have unit length:  $\|a\|^2 = 1$ , a sphere is defined by a centre and radius and so on. We may expect, then that manipulation of primitives depends to some extent on manipulation of parameters.

Putting together these ideas with those from the last section we can arrive at some basic principles.

1. A *constraint region* is a set which can be built from elementary primitives by Boolean operations.
2. It is often convenient to describe a primitive as a level or sub-level set of a parametrized function:  $g(x, a) = b$  or  $g(x, a) \leq b$ .
3. There may be complexity reductions in performing Boolean operations on primitives.

## 6 Mapping constraints

Assume that the function  $f(x)$  is known. Then mapping of constraints governed by the following simple rules:

$$\begin{aligned} f(A \cup B) &= f(A) \cup f(B) \\ f^{-1}(A \cup B) &= f^{-1}(A) \cup f^{-1}(B) \\ f(A \cap B) &\subseteq f(A) \cap f(B) \\ f^{-1}(A \cap B) &\subseteq f^{-1}(A) \cap f^{-1}(B) \end{aligned}$$

If  $f$  is unknown but lies in some known class:  $f \in \mathcal{F}$ . Then at least we can say that

$$\begin{aligned} \mathcal{Y} &\subset \cup_{f \in \mathcal{F}} f(\mathcal{X}) \\ \mathcal{X} &\subset \cup_{f \in \mathcal{F}} f^{-1}(\mathcal{Y}), \end{aligned}$$

which may give us information about the shape of induced constraints. If, for example,  $\mathcal{F}$  is the set of  $f$  within some confidence band then we may propagate the band through  $f$  and obtain an output bound. Conversely the band may

actually *define* the constraint region  $\mathcal{Y}$  of  $y = f(x)$  and we can propagate backwards with  $f^{-1}$ . Inverse (also reverse) engineering methods are another source of literature on such methods.

## 7 Ellipses etc

A famous problem which has been studied for over 50 years is that of finding the minimal volume ellipsoid which contain a set of point in  $R^p$ . It turns out that this is the dual of a D-optimum design problem, but we do not discuss this further here. The maximum volume interior ellipse problem became popular as part of the machinery of the Khachiyan-Todd algorithmi for fast interior point method alternatives to the traditional simplex algorithm in linear programming. There are important dualities between the inner and out problems and fast algorithms exist for both. [1]

We can also consider minimal volume hyper-rectangle, or other regions and set them up them up as optimisation problems. We can go further and follow the union of intersections idea and take extended regions formed of unions of ellipsoids or other figures. Covering a difficult region such as a boundary with the union of balls, ellipses or boxes (hyper-rectangles) has been used in a number of field. An interesting use is in computing fractional dimensions in the theory of fractals and dynamical systems; hence the name box-dimension.

### 7.1 Constraint management in MUCM

We can now give a loose definition of constraint management as the inputting, storage, manipulation, updating and outputting of constraint regions and their associated primitives. We can expect that some systems, such as those arising in data-bases management and similar fields, to be logical in nature, using advanced ideas from computer science and Boolean algebra and others, although using logic, will more geometric in flavour. We can also expect to find complexity reduction within the methods although they may be sometimes hidden. There is a heuristic field of constraint management which takes into account cost and many other types of commercial and operational constraints. It may be that such constraints affect geometric constraints, but we will not discuss this issue further here.

We collect together the main features and terminology of practical constraint management systems.

1. Constraint region, also: feasible region, operability region, safe domain etc.
2. Feasible and infeasible: when a set of  $x$  or  $y$  value lie in the constraint region or not. Checking feasibility is a key operation.

3. Dynamic constraints: those that change over time. It may refer to when *new* constraints arrive or old ones disappear. One can talk about *active constraints*, *constraint acquisition* etc.
4. Query. An operation which returns all the *instances* of *class*, in a discrete problem. In our terminology we output all feasible examples of some particular constraint set.

The problem that needs to be addressed is what type of constraint management might be of use in MUCM and what kind of problems it might be useful for.

It seems to us that the main reasons for some consideration of constraint management can be give:

1. Input and output regions may be complex and occupy small volume
2. There is a need for experimental design to be similarly constrained
3. Given that the output may be constrained it will be important to describe the implied constraint on the input
4. Data assimilation may need to take account of constraint assimilation
5. Covariance functions may be spatially dependent and constraints may be needed to partition the region: eg, the use of Voronoi diagrams
6. Optimisations problem may be constrained

## 8 Two methods for experimental design

*Rejection* The most obvious way of restricting an experimental design to a constraint region  $\mathcal{X}$  is to use *rejection*. Let  $D$  be a design of fixed sample size or a procedure for selecting design points on a simple region  $\mathcal{X}_0 \supset \mathcal{X}$ . Then select all those members of  $D$  which lie in  $\mathcal{X}_0$ , which may produce a sample size not known in advance, or run a sequential procedure and carry on selecting points in  $\mathcal{X}$  until a required sample size is achieved.

The act of selection feasible design points is also the act of rejecting infeasible design points. Each new point, in the sequential method, gives yields a query. If the constraint region  $\mathcal{X}$  is an intersection of primitives such as a region  $\{x : g(x) \leq c\}$  then one can check each primitive one by one and only accept if each is feasible. If there is a more complex structure then, as in data-base querying, special procedures may be available to speed up the query. The discussions in Sections 3 and 4 may be relevant. For an approximate method, or for speeding up a rejection method, one may enclose the constraint region in a simpler region such as ellipse and use that as a

coarse rejection step. There are special methods for testing feasibility such as in LP.

*Variable geometry.* Suppose again we have a simple region  $\mathcal{X}_0$  and let the true constraint region be a transformation of this:  $\mathcal{X} = G(\mathcal{X}_0)$ , where  $G$  is 1-to-1. Then  $x \in \mathcal{X}$  is equivalent to  $G^{-1}(x) \in \mathcal{X}_0$ . In this case we can start with a design  $D_n^{(0)}$  on  $\mathcal{X}_0$  and use the design  $D_n = G(D_n^{(0)}) \subset \mathcal{X}$ . If  $G$  is linear and uniform then  $D_n$  may be similar, although some might be affected. For optimal design there is work to do in converting an optimisation problem on  $\mathcal{X}$  back to one on  $\mathcal{X}_0$ . Clearly in the kriging case, covariance and optimisation criteria will change under  $G$ .

## References

- [1] Okabe, A. Boots, B and Sugihara, K (1992). Spatial tessellations: concepts and applications of Voronoi diagrams. Wiley, New York.
  - [2] Sun, P. and Freund, R. M. (2004). Computation of Minimal-volume Covering Ellipsoids. *Operational Research*. 52, 690–706.
  - [3] Christiani, N and Shaw-Taylor, J. (2000). An introduction to Support Vector machines and other kernel-based learning methods. Cambridge University Press.
  - [4] Naiman, D. Q. and Wynn, H. P. (1997). Abstract tubes, improved inclusion-exclusion identities and inequalities and importance sampling. *Ann. Statist.* 25. 1954–1983.
  - [5] Naiman, D. Q. and Wynn, H. P. Inclusion-exclusion-Bonferroni identities and inequalities for discrete tube-like problems via Euler characteristics. *Ann. Statist.* 20 (1992), 43–76.
  - [6] Edelsbrunner, H and Koehl, P. (1995). The union of balls and its dual shape. *Discrete and Copmut. Geom.* 13, 415–440.
  - [7] Stanley, P. (2004). An introduction to Hyperplane arrangements. <http://www.math.umn.edu/~ezra/PCMI2004/stanley.pdf>
  - [8] Adler, R, J and Taylor, J. (2007). Random fields and geometry. Springer-Verlag, New York.
- [Note. More references as this review develops]