# Design as a Marked Point Process

John Quigley[1], Gokula Vasantha[2], Jonathan Corney[3], David Purves[1], and Andrew Sherlock[4]

[1]Department of Management Science, University of Strathclyde, G1 1XQ, Scotland

[2]School of Engineering and the Built Environment, Edinburgh Napier University, EH10 5DT, Scotland

[3]School of Engineering, University of Edinburgh, EH8 9YL, Scotland

[4]National Manufacturing Institute Scotland, University of Strathclyde, G1 1XJ, Scotland

**Abstract**

Although AI systems which support composition using predictive text are well established there are no analogous technologies for mechanical design. Motivated by the vision of a predictive system, that learns from previous designs and can interactively provide a list of established feature alternatives to the designer as a design progresses, this paper describes the theory, implementation and assessment of an intelligent system that learns from a family of previous designs and generates inferences using a form of spatial statistics.

The formalism presented, models 3D design activity as a 'Marked Point Process' that enables the probability of specific features being added at particular locations to be calculated. Because the resulting probabilities are updated every time a new feature is added the predictions will become more accurate as a design develops. This approach allows the cursor position on a CAD model to implicitly define a spatial focus for every query made to the statistical model. The authors describe the mathematics underlying a statistical model that amalgamates the frequency of occurrence of the features in the existing designs of a product family.

Having established the theoretical foundations of the work, a generic six step implementation process is described. This process is then illustrated for circular hole features using a statistical model generated from a dataset of hydraulic valves. The paper describes how the positions of each design's extracted hole features can be homogenized through rotation and scaling. Results suggest that within generic part families (i.e. designs with common structure) a marked point process can be effective at predicting incremental steps in the development of new designs.

*Keywords*: Feature based Design, Predictive Design, Marked Point Process

# 1 Introduction

It has been argued that only 20% of design information is reused despite 90% of all design activities being based on the variants of existing designs [1], and that on average only 28% of design information is reused within manufacturing applications [2]. Design can be considered as a sequential decision-making process, where the current state of a design evolves through a series of design choices. A system is required where design features may be suggested to the designer for effective reuse, and these design reuse procedures can be learned from historical data [3, 4].

This paper introduces the underpinning mathematics required for implementation of a new generation of user interfaces that automatically identifies appropriate characteristics of previous designs for reuse based on a designer's real time activity. As a design evolves the system generates predictions of the features which might be incorporated, and are informed by both previous work and the new, ongoing design. In order to identify the most relevant features, and avoid presenting the user with an overwhelming number of suggestions, the work reported exploits the location of information (i.e. features and mouse pointer) on a 3D Computer Aided Design (CAD) model so that predictions can be appropriate to specific positions on an engineering component. The system described assumes a single engineer developing a design by carrying out a series of operations on a CAD system. The system does not dictate any order of operations and allows the engineer's focus to move around the component.

Designs seldom start with a blank sheet of paper, but are informed by past experiences with reports of as much as 75% of design activity comprising the re-use of existing knowledge [5]. In the context of designing industrial parts such activities comprise re-using, configuring, and assembling existing components. A key contributing factor to companies not performing projects on time and budget is the lack of knowledge re-use, which leads to frequent 'reinventing the wheel' rather than finding and using already known solutions [6].

Motivated by these observations this paper proposes a different form of design representation that can combine many design variations into a single probabilistic model that facilitates the reuse of previously used features during an interactive process that leads to the instantiation of a new design. By leveraging the available information, a probabilistic CAD system would prompt the engineers with fragments (i.e. features) of previously designed components to extend the current CAD design. Although reuse of common features in the design of many industrial products is desirable there could be cases where such a practice inhibits innovation. Aware of this the authors' aim is not to automate but support with suggestions that the engineer is free to ignore. For this we propose modeling the design process as a Marked Point Process (MPP) to create a formal framework that can assess the association between designs. Similar approaches have been used successfully in neuroanatomy to analyze brain scan images through voxel based morphometry [7], as well a feature recognition in image analysis.

For our application, points are the coordinate location corresponding to where a design feature has

been placed and marks refer to the feature chosen. MPP is a form of spatial statistics, metrics based on statistical tools that are used to characterize the distribution of events across space [8], and are widely used across a number of application areas for example the distribution of trees in forests to stars in the sky. Through this lens we view the behavior of engineers as a stochastic process, updating throughout the design process on decisions made to place features in specific locations and thereby supporting probabilistic measures for subsequent choices. The statistical inference can be supported through historical data, viewing past designs as realizations from such a stochastic process. Specifically, we develop a decision support system through a Bayesian methodology, where we start with a prior distribution to assign a probabilistic measure on the features and location to be chosen by the engineer. Following each choice, the prior distribution is updated to a posterior distribution based on this new data and thereby making full use of all the information available. So as more design choices are made, the model will be able to discriminate more effectively between historical designs based on similarity.

Given the above context and motivation, the authors defined the following goals for the work:

## Aim

To define a computational framework that can support an interactive design process with suggestions of features based on three inputs: a knowledge of existing designs; the state of an emerging design and a location on the surface of the emerging design.

## Objectives

1. Establish a method of homogenizing the orientation and dimensions of a collection of designs belonging to a product family.
2. Develop a statistical function that represents the probability of a particular feature occurring at a particular location, on the surface of a design for a member of the product family.
3. Create a prototype implementation that can support an interactive design cycle which updates the inferred probability of specific features occurring at given location as the design of a part is modified.
4. Assess the accuracy of the feature predictions
5. Identify any inherent limitations or weakness in the approach

The rest of this paper is structured as follows: In Section 2 we provide a brief review of both predictive design systems and relevant MPP literature to position the contribution of this work. In Section 3 we present a generic overview of our process to support design development and in Section 4 we outline the details of the mathematical model that underpins the process. We explore key characteristics of design

activities and data to inform our modeling choices, and we provide a generic modeling and decision support framework. In Section 5, we evaluate the proposed modeling framework through a case study. Finally, in Section 6 we reflect on the future direction of research in this area.

## 2 Literature Review

Systems for predictive design have to combine assessment of historical data with statistical methods so that human users can easily choose, or ignore, suggestions that enhances the creative process. SMS text messaging software, used by mobile phones, illustrate both the potential and challenges of engineering useful predictive systems. However, while text prediction seek to identify patterns in a linear series of symbols with a simple (i.e. keypad) interface, anticipating the intention of product designer requires analysis of 3D information that has no canonical ordering (i.e. unlike a sentence of text, that reads from left to right, a designer can essential edit shapes in any sequence). Despite these inherent difficulties research into computational technologies that could enable predictive design systems has been reported for more than a decade.

An early example is [9] who developed the "InspireMe" interface which allowed a user to 'place' and 'glue' one of ten suggestions, proposed in response to a query shape, and then request new suggestions for the resulting composite shape. The placed shape can be translated, rotated, and scaled to match the query shape. The suggestions that are not useful can be removed and replaced with new suggestions. [9] used a multi-dimensional histogram-based signature to encode shape's global spatial structure and its local detail to identify suggestions for a given shape query.

Later work recognized that there was potential to improve the accuracy of suggestions by combining the frequency of occurrence with shape parameter values. For example, [10] demonstrated an interface for an assembly-based modeling tool. The interface presents the user with semantical labeled tabs that can be expanded hierarchically to show component sub-categories. The user can select a component and drag it onto the current model. A probabilistic Bayesian network is then used to dynamically update both the proposed component categories and the components based on their semantic and stylistic compatibility with the current modeling state. The interface estimates whether the new component should have a symmetric counterpart and computes the symmetry plane. Based on the modeling requirements, the selected model can be moved to a position, rotated, scaled, duplicated, and glued.

While [9] focused on the design of assemblies of predefined component parts, [11] reported a predictive system for component shape synthesis. Their approach provided an interactive platform for the user to constrain shape synthesis based on high-level specifications (i.e. specific components, components from particular categories, and components from learned latent styles) and an input shape database. Within the proposed interactive shape synthesis interface a user can select constraints by selecting required: shape

styles, component categories and styles. The algorithm then proposes a list of synthesized objects based on the given inputs. The discrete features help ensure that components selected for a synthesized shape have compatible numbers of adjacent components of each type, and their edges have been identified and stored with the category label of components (so they can be attached for placing where a component can be attached to another component with symmetry relationships). Like [9], [11] also used a probabilistic approach to identify and synthesize existing shapes from complex domains to generate new combinations of components.

There is a tension in all reported work between accuracy and the number of predictions made. This can be observed in [12] that describes a user interface that guides a designer's selection with a list of the 50 'best' suggested components during an assembly based modeling process. The interface aims to enable easy browsing and propose components that are most compatible with the current state of the assembly design (represented as a 3D model). The interface allows users to manually drag, move, scale, orient and combine selected components. The placed components can also be incorporated in the design using Boolean operations (union, difference, and intersection) to obtain composite model. The suggestion list automatically updates every time a component is added to the assembly. The suggestions are ranked by size (larger components are given preference) at the start of the modeling process. The marginal probability distribution computed from a factor graph by [12], which incorporates adjacency and multiplicity factors of segmented components, to score and rank predicted components.

A different type of assembly design is considered by [13] who proposed an algorithm that takes a partially completed 3D scene as input and propose relevant models in a user-specified region of interest by leveraging text data. Suggestions are generated using three different approaches; Graph Kernel, N-gram, and Merged. A query is generated by converting the given 3D scene into text that represents the five closest models to a focal point nominated by the user. The algorithm uses co-occurrence, 5-gram statistics from Google Web N-grams dataset and point-wise mutual information between the labels of nearby models in the scene and the labels of models in the database to create suggestions.

For a very similar application [14] presented a method for generating novel arrangements of diverse 3D objects synthesized from few given examples. The method creates a probabilistic model for scenes based on Bayesian networks and Gaussian mixtures that can be trained by a small number of input examples of relevant scenes retrieved from database. User were able to vary the degrees of similarity and diversity in the generated scenes by controlling the weighting (through blending parameters) given to the influence of the existing database of prior designs.

The "AttribIt" interface was developed in [15] which facilitates the targeted exploration of different combinations of visual components using commands based on the relative semantic attribute. A user initializes a design with a coherent combination of components from a database, then they select a subset of these components and interactively increase, or decrease, the strength of an attribute using sliders.

In doing this they can observe changes to the whole design in real-time as new database components corresponding to the updated attribute strengths are swapped. The components are assembled automatically into a coherent design (provision for manual adjustments such as translation, rotation and scaling controls are available to refine the results). The interface shows regions of high geometric variation under the current attribute (highlighted in red color in Fig. 1).


Figure 1: User Interface for Assembly-based modeling using relative attributes [15]

When the overall form of a design (whether assembly or component) is constrained by function or the need to fit into a product family a template can be used to facilitate reuse. For example [16] developed templates that can be used in an interactive design system to create new 3D models in a design-by-example manner. The interface allows a user to choose template parts from the database, change their parameters, and combine them to create new models. The information in the template has been used to automatically position, align and connect parts by adjusting parameters, adding constraints, and assigning connectors. The assembly-based modeling system provides pick and drag substructures from different designs and add them to a working model. The elements on the selected node are represented in full color, while the others become semi-transparent during manipulation, and constrained degrees of freedom are hidden.

To support the generation of interior designs [17] developed a probabilistic hierarchical grammar to enable functional (rather than spatial) representation of an office environment. The aim was to support consistent segmentations, category labels and functional groupings of 3D scenes that characterizes geometric properties, cardinalities and spatial relationship in a hierarchical manner. A probabilistic grammar is used to automatically create consistent annotated scene graphs. Figure 2 illustrates an input scene mapped with labels and then converted into the hierarchical form using probabilistic grammar. A 7-dimensional descriptor (i.e. support and vertical relationships, horizontal separation, and overlap between objects) is used to describe the relationship between two objects. The dynamic programming for belief propagation was developed for scene parsing with optimal hierarchy. The technique creates candidate nodes based on spatial proximity, grammar binaries and finds the optimal binary hierarchy

which is converted to a logical hierarchy of the original grammar.

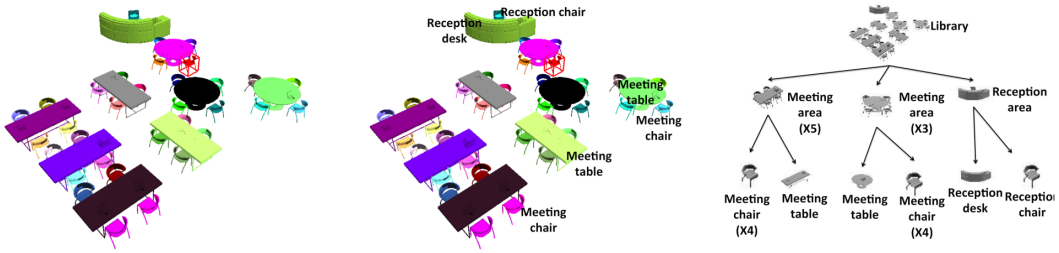



Figure 2: Input scene mapped with labels and converted into the hierarchical form using probabilistic grammar [17]

More recently [18] reported the "ComplementMe" user interface that aims to seamlessly integrate suggested CAD models into the design process. A combination of embedding and retrieval neural network architectures are proposed for suggesting complementary functional and stylistic components and their placements within an incomplete 3D part assembly. The embedding network was used to map parts to a low-dimensional feature space, and the retrieval network was used to retrieve partial assemblies to appropriate components. The interface shows the possible candidates generated by sampling from the conditional probability distribution predicted by the retrieval network. The user could select a desired complementary component, and the algorithm predicts the location for it via the placement network. The new shape will be synthesized for the user, and the next component is proposed based on the modified assembly.

In conclusion the literature on predictive design systems is largely focused on the creation of assemblies of 3D component models where frequently the positioning of suggested components is a manual task for the user. In contrast the authors' work is focused on the identification of shape features (i.e. fragments of an entire model that are patterns of geometry such as holes) that are appropriate to a location defined by the position of a user's mouse pointer on the surface of a 3D object.

## 2.1 Marked Point Processes

Marked point processes (MPP) are widely applied within image analysis, where it was first introduced by [19]. The methodology is used extensively and successfully for the extraction of multiple objects from images. Applications include biological imagery on cells [20], disks in a plane [21], building outlines [22] and person detection from camera images [23]. It is a flexible methodology that has been extended for object extraction from images to arbitrarily shaped objects [24]. More recently, [25] have developed the approach for microscope images, [26] have used MPP's to automatically detect the locations of road segments and [27] have used it for visual perceptions. A survey of marked point processes applied to image analysis can be found in [28].

The literature to date has developed methods to extract images and characterize them in the form of a MPP which are then stored in a database. Our focus complements this work, as we develop decision

support tools that also utilizes information about the location of extracted features in an MPP data structure.

# 3 Process for Constructing Marked Point Process Decision Support

We propose a six step approach adapted from the CISSE process, see [29], for constructing empirical prior distributions to support Bayesian analysis, which considers the following five steps; *Characterize*, *Identify*, *Sentence*, *Select*, and *Estimate*. As described in the following for the third step we have placed particular focus on homogenizing the data rather than sentencing the data and we have decomposed the fifth step to consider prediction and updating.

Step 1: *Characterize the population of designs.* We begin by identifying those factors characterizing the design. This is an important step because it defines the criteria by which data sets (i.e. historical designs) are subsequently selected for inclusion in the comparator pool used to construct the prior distribution. Examples of such characteristics may be with respect to types of layouts of and/or features used within designs.

Step 2: *Identify candidate sample designs matching population.* The factors characterizing the population of designs can be switched on/off for candidate designs effectively providing a means of making a relative assessment of relevance against a set of criteria. We are simply trying to find the best available data sets to make reasonable and timely inference. We are assuming that the current design for which we are providing the decision support will be similar to one of these historical designs. We can accommodate a unique apriori assessment on the likelihood of the current design being realized to be like each possible candidate historical design, although our default may be a uniform distribution prior.

Step 3: *Homogenizing the comparator data.* Generally, the higher the degree of homogeneity within the comparator pool the more accurate the predictive inference [see 30]. This requires a measure for similarity between designs, such as the KL divergence measure as proposed in [31] against which the data can be transformed for homogeneity. Two key approaches to address this are scaling and rotation. Firstly, all designs can be re-scaled into the unit cube. Secondly, the data describing the locations of features can be rotated for alignment. This work should be performed prior to the start of the design. This stage may be omitted if it is considered that information would be lost in transforming the data, and the resulting prior would not be as effective at discriminating between design types.

Step 4: *Select a probability model for the population of designs.* The nature of design patterns is such that a parametric probability distribution is unlikely to exist that adequately represent the variability of location and features within designs. As such, a non-parametric approach should be considered, for which we recommend Kernel Density Estimation (KDE). Under such an approach, choices will need to be made concerning the band-width parameter, which is essentially deciding on allowable variation of location of features within similar designs. The resulting model is known as the Feature Location Probability Function (FLPF), for which we would fit one to each historic design to obtain a model for each design type.

Step 5: *Predictive Model.* The predictive distribution is simply a weighted average of the FLPF for each design type in the comparator set, where the weights reflect the likelihood that the current design will ultimately be realized as being similar to the candidate design in the set.

Step 6: *Update prior on design type and predictive distribution.* During the design process, Steps 5 and 6 are repeated in a cycle of feature addition and updating of the predictive distribution, which we call the Predictive Feature Location Function (PFLF), to reflect how each change impacts on the probable location of other features. This process, driven by the actions and selections of the human designer, continues until the component part is complete (i.e. the design is finished).

Figure 3 provides a schematic for the predictive system. The data homogenization and FLPF can be performed in advance using the existing designs and features selected from the database in steps 1 and 2. As a new design evolves, the PFLF is generated from the FLPF and the design type prior. The PFLF can be updated in response to events to provide feature suggestions at interactive speeds.

# 4 Model Development

In this section, a model is mathematically developed for steps 3 to 6 from the process in Section 3. This will allow for both predictions on feature type together with its spatial position.

## 4.1 Overview

We model the process of a designer choosing to place features in specific locations as a Marked Point Process (MPP). As such we can view historical designs as a realization from this process. Consider a design denoted by $d_i$, which comprises $n_i$ features (not necessarily unique) and for each feature, which we denote with $m$, we have an associated location described by its $(x, y)$ coordinates. We express the design as an unordered set of coordinates and features with $d_i = \{(x_1, y_1, m_1), \ldots, (x_{n_i}, y_{n_i}, m_{n_i})\}$. We restrict our designs to 2 dimensions expressed as $(x, y)$ coordinates for simplicity but the method is easily generalizable to higher dimensions.

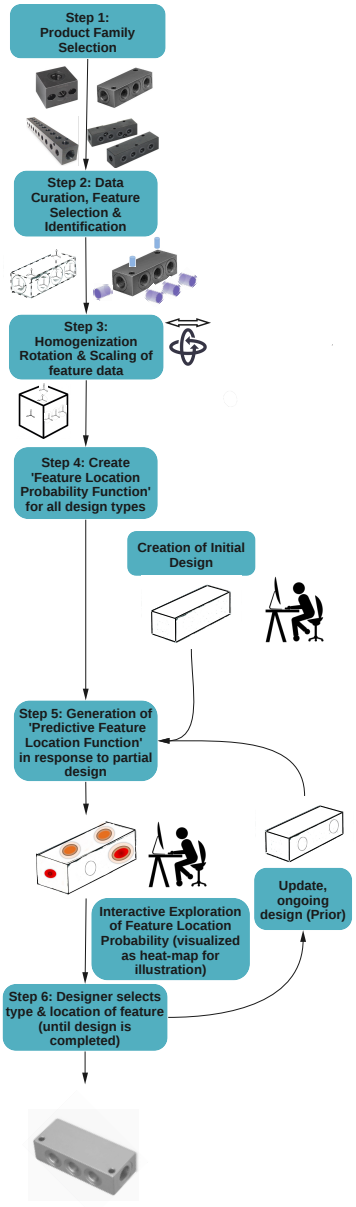



Figure 3: System schematic for feature prediction through Bayesian updating.

At the core of a MPP is the intensity function, which describes the probability of a feature being placed in a particular location. Let $\lambda(x, y, m|\, \underset{\sim}{c})$ denote the intensity function of the process at location specified by the $(x, y)$ coordinates, for feature $m$ given the designer has already made choices of places various features at locations captured in the matrix $\underset{\sim}{c}$. A characteristic of this function is that if we integrate the intensity over the whole $(x, y)$ plane then we obtain the expected number of features $m$ in the design. Moreover, we can express this as a probability density function, i.e. $f(x, y, m, |\, \underset{\sim}{c})$ given in Eq. (1), to describe the next choice made by the designer by normalizing it so that it integrates to 1. This function can then be used to rank features based on their likelihood of being placed at specific locations to provide appropriate decision support to the designer.

$$f(x, y, m, |\, \underset{\sim}{c}) = \frac{\lambda(x, y, m|\, \underset{\sim}{c})}{\iint\limits_{\forall x,y} \lambda(x, y, m|\, \underset{\sim}{c}) \mathrm{d}x\mathrm{d}y} \tag{1}$$

Engineering designs possess dependency structures unlike other fields of MPP study so 'off the shelf' models for intensity functions are not available. Dependency refers to the association of choices, such that placing one feature in a location increases or decreases the likelihood of other features in various locations. Poor choices of dependency models can result in uninformative inference at best and misleading inference at worst. In typical spatial or temporal point process applications, self-exciting models are used to capture local dependency where the realization of one point increases the likelihood of nearby points being discovered. In design, choosing a feature for a location can have ramifications for distant locations due to a need for symmetry for example. We develop a methodology for characterizing such dependency.

Many designs may be a collection of few choices, so while there may exist a large database of historical designs there are small sample sizes on which to infer the dependency structure. Inference is made more challenging with an extensive set of features from which to choose.

We propose a non-parametric approach to estimating the intensity functions that will provide a foundation on which to develop decision support, estimated from the data on historical designs. Kernel density estimates (KDE) consists of modeling the intensity function of a point process through assigning a kernel, e.g. the Normal distribution, centred at each location where a point has been realized, often resulting in a multi-modal probability model to describe the likelihood of discovering points. Typically the kernel density requires the analyst to choose a value for the smoothing parameter (e.g. in the case of the Normal kernel density this would correspond to the standard deviation for each density used).

In Section 4.2 we will develop the non-parametric model for the density function based on KDE from historical designs. In Sections 4.3 and 4.4 we will outline a Bayesian updating mechanism that will show how the density function changes as the designer makes further choices and as such so too will the decision support. In Section 4.5 we will derive metrics to characterize the dependency structure implied by these modeling assumptions. Finally in Section 4.6, we will consider transformation that we can make

on historical design data to improve predictions, specifically, re-scaling and rotating the data.

## 4.2 Model Description

We assume that a new design will be similar in some sense to historical designs but not necessarily identical. As such, prior to commencing an assessment should be made of the historical data that will be used to assess its suitability. Assuming we have a catalog of $n$ historical designs that are appropriate for the decision support then we consider that there are $n$ types of design and the current design under construction will belong to one of these types. We will estimate the density function for each type with the data available from each design. Following this we will apply a prior probability on the type of design being constructed based on the choices made.

Consider an historical design $i$ for which there have been $n_{i,m}$ choices of feature $m$. Using a KDE approach to estimate the probability density function for design of type $i$ with respect to feature $m$ we have the density given in Eq. (2)

$$f(x,y|m,i) = \begin{cases} \frac{1}{c_{xy}} & n_{i,m} = 0 \\ \frac{\sum_{i=1}^{n_{i,m}} \phi_j(x,y;\mu_{x,j} = x_{i,m,j}, \mu_{y,j} = y_{i,m,j}, \sigma)}{n_{i,m}} & n_{i,m} \geq 1 \end{cases} \tag{2}$$

Where $\phi_j(\cdot)$ is a bivariate Normal density function, $\mu_{x,j}$ is the mean of the $x$ variable in $\phi_j(\cdot)$, $\mu_{y,j}$ is the mean of the $y$ variable in $\phi_j(\cdot)$, $x_{i,m,j}$ is the location on the $x$ coordinate in design $i$ of the $j^{th}$ occurrence of feature $m$, $y_{i,m,j}$ is the location on the $y$ coordinate in design $i$ of the $j^{th}$ occurrence of feature $m$, $\sigma$ is the standard deviation for both $x$ and $y$, although one could assume a more elaborate covariance structure if appropriate, and $c_{xy}$ is a normalizing constant to ensure the density integrates to 1. It is worth noting that one could substitute other kernel density functions in if more appropriate, we only require it to possess all the characteristics of a bivariate probability density function.

We have assigned a uniform distribution over the plane for situations where that feature has not appeared in design $i$. It may be desirable to remove this, if one did not want to permit certain features for particular design types.

Essentially, the resulting density is a collection of Normal densities centred about observed locations and the standard deviation parameter controls for the allowable variation from the historical design to be considered similar.

Let $I$ be the random variable describing the design type that the designer is developing and $M$ to be the random variable describing the next feature to be chosen. To express the unconditional probability density function we first define three indicator functions to denote design type, feature, and presence in Eq. (3).

$$\delta_i = \begin{cases} 1, & I = i \\ 0, & I \neq i, \end{cases} \qquad \delta_m = \begin{cases} 1, & M = m \\ 0, & M \neq m, \end{cases} \qquad \delta_{n_{im}} = \begin{cases} 1, & n_{im} \geq 1 \\ 0, & n_{im} = 0 \end{cases} \tag{3}$$

We denote the probability of a feature $m$ appearing in design type $i$ with $p_{i,m}$ and the probability of the design being of type $i$ with $\pi(i)$. Combining these, the full probability density function describing the likelihood of a feature $m$ being located at $(x, y)$ and the design being of type $i$ is given in Eq. (4).

$$f(x, y, M = m, I = i) = \sum_{i=1}^{i_{max}} \sum_{m=1}^{m_{max}} \delta_i \pi(i) \delta_m p_{i,m} \left( \delta_{n_{im}} \frac{\sum_{j=1}^{n_{i,m}} \phi_j(x, y; \mu_{x,j} = x_{i,m,j}, \mu_{y,j} = y_{i,m,j}, \sigma)}{n_{i,m}} + (1 - \delta_{n_{im}}) \right) \tag{4}$$

The design type is a latent variable used to capture the dependency between the features and locations. By summing the density function across all possible values of $I$ we obtain the distribution for location and feature only, given in Eq. (5).

$$f(x, y, M = m) = \sum_{i=1}^{i_{max}} \sum_{m=1}^{m_{max}} \pi(i) \delta_m p_{i,m} \left( \delta_{n_{im}} \frac{\sum_{j=1}^{n_{i,m}} \phi_j(x, y; \mu_{x,j} = x_{i,m,j}, \mu_{y,j} = y_{i,m,j}, \sigma)}{n_{i,m}} + (1 - \delta_{n_{im}}) \right) \tag{5}$$

Similarly, we express each marginal distribution in Eqs. (6–9).

$$f(x) = \sum_{i=1}^{i_{max}} \sum_{m=1}^{m_{max}} \pi(i) p_{i,m} \left( \delta_{n_{im}} \frac{\sum_{j=1}^{n_{i,m}} g_j(x; \mu_{x,j} = x_{i,m,j}, \sigma)}{n_{i,m}} + (1 - \delta_{n_{im}}) \right) \tag{6}$$

$$f(y) = \sum_{i=1}^{i_{max}} \sum_{m=1}^{m_{max}} \pi(i) p_{i,m} \left( \delta_{n_{im}} \frac{\sum_{j=1}^{n_{i,m}} g_j(y; \mu_{y,j} = y_{i,m,j}, \sigma)}{n_{i,m}} + (1 - \delta_{n_{im}}) \right) \tag{7}$$

$$f(M = m) = \sum_{i=1}^{i_{max}} \pi(i) p_{i,m} \tag{8}$$

$$f(I = i) = \pi(i) \tag{9}$$

Where $g_j(\cdot)$ is a univariate Normal density function.

## 4.3 Probability of a Feature Being Selected for a Design

Given the total number of incidences within a design we assume the number of incidences of each possible feature for a design is a realization from a multinomial distribution. Moreover, we assume the underlying probabilities associated with each feature vary across design types. Under such a modeling assumption a natural estimator of the probability of a feature being selected for a design of a particular type would be the observed frequency on similar designs from the class. However, given that we have at most one design for each type we are likely to produce poor inference due to small samples. Moreover, we are likely to be faced with a large number of features with zero events data resulting in an estimated probability of 0. This creates a particular issue for the decision support being developed, as all historical designs that did not possess all the features chosen for a current design would be ruled out as candidate design types through Bayesian updating. As such, allowing for non-zero probability estimates would permit the inclusion of candidate design types even if they do not include all the features chosen at some point in the design process. For a discussion on alternative estimation methods for zero event data, see [32].

We propose using an uninformative prior distribution, where a uniform prior distribution is assumed on each probability and subsequently updated on the data. As the probabilities must sum to 1, the uniform assumption implies a Dirichlet prior distribution. This is a common pairing with the multinomial distribution as it provides a flexible distribution that is convenient to use computationally. This results in the following estimate for the probability, given in Eq. (10).

$$p_{i,z} = w \frac{\beta_z}{\beta} + (1 - w) \frac{k_{i,z}}{k_i} \tag{10}$$

Where $w = {}^{\beta}/_{(\beta+k_i)}$, $k_{i,z}$ gives the number of features in design $i$ of type $z$, $k_i = \sum_{\forall z} k_{i,z}$, $\beta_z = \sum_{\forall i} k_{i,z}$, and $\beta = \sum_{\forall z} \beta_z$.

We see that $p_{i,z}$ is a weighted average of the observed frequency ${}^{k_{i,z}}/_{k_i}$ and the prior mean. The weight applied to the frequency increases as the number of features chosen for design $i$ increases, i.e. $k_i$.

## 4.4 Bayesian Updating

Every choice made by the designer provides information concerning the type of design being constructed, i.e. to which historical design is it similar. We will model this learning through Bayesian updating. As described in Section 4.2 we have a probability distribution, i.e. $\pi(i)$, which describes the uncertainty concerning the design type. In this section we present a Bayesian updating of this distribution based on design choices. Assume that the designer has made $n_k$ choices then the posterior distribution for the design type is updated as in Eq. (11).

$$\pi(i|\underset{\sim}{c}) = \frac{\prod_{k=1}^{n_k} f(x_k, y_k, M = m_k, I = i)}{\sum_{i=1}^{i_{max}} \prod_{k=1}^{n_k} f(x_k, y_k, M = m_k, I = i)} \tag{11}$$

This posterior is then used in the predictive distribution, given in Eq. (12).

$$f(x, y, M = m|\underset{\sim}{c}) = \sum_{i=1}^{i_{max}} \sum_{m=1}^{m_{max}} \pi(i|\underset{\sim}{c}) \delta_m p_{i,m} \left( \delta_{n_{im}} \frac{\sum_{j=1}^{n_{i,m}} \phi_j(x, y; \mu_{x,j} = x_{i,m,j}, \mu_{y,j} = y_{i,m,j}, \sigma)}{n_{i,m}} + (1 - \delta_{n_{im}}) \right) \tag{12}$$

This function can be used to provide inference on the relative likelihood of features being located on specified positions through comparing ratios.

## 4.5 Dependency Structure

The moments of the model are easily obtained through conditional expectation arguments resulting in the expectations given in the Supplemental Material S1. Through setting $\delta_{n_{im}} = 1$ for all designs we would obtain the moments anticipated in the historical designs, however, for our model we have accommodated the possibility of features appearing in design types which are not present in the associated historical design.

The moments can then be used to construct measures such as correlation between the $(x, y)$ coor-

dinates. However, while commonly used such measures are limited within our context as they focus on the linear relationship between only two variables. We may wish to consider more general settings such as non-linear relationships as well as 3D designs or even the dependency between features and locations. For this we use the mutual information (MI) measure, which we denote by $\omega$, to assess dependency. The concept of mutual information is linked to the entropy of a random variable, which quantifies the expected amount of information held in a random variable. The mutual information measure is considering the information gain from modeling the joint distribution rather than assuming each variable is independent.

$$\omega = \mathbb{E}\left[\ln\left(\frac{f(X,Y,M)}{f(X)f(Y)f(M)}\right)\right] \tag{13}$$

This can be re-expressed as in Eq. (14), which is simply the entropy of the joint distribution minus the sum of the entropy for all the marginals.

$$\omega = \mathbb{E}\Big[\ln\big(f(X,Y,M)\big)\Big] - \mathbb{E}\Big[\ln\big(f(X)\big)\Big] - \mathbb{E}\Big[\ln\big(f(Y)\big)\Big] - \mathbb{E}\Big[\ln\big(f(M)\big)\Big] \tag{14}$$

Clearly, $\omega = 0$ if $f(x,y,m) = f(x)f(y)f(m)$, i.e. if the variables are independent. Moreover, it can be shown that as dependency increases so too does the measure. This measure can be useful for comparing dependency between various subsets of designs, noting that the stronger the dependency the better the predictions will be. Some analysts prefer to transform this measure to bound it within $(0,1)$ and as such use the transform $\dot{\omega} = \sqrt{1-e^{-2\omega}}$ [33]. The joint and marginal distributions required to calculate the MI are provided in S2.

## 4.6 Re-Scaling and Rotating

Generally, the higher the degree of homogeneity in the comparator pool of data, then the greater the accuracy in the prediction [30] and as such pre-processing the relevant historical data to achieve greater homogeneity may be desirable. We consider re-scaling and rotating the data for each as a means to achieve this. However, such transformations may not always be beneficial as key information may be lost that helps identify the most similar historical designs. The advantage of such transformations are through identifying regions where specific features are highly likely to be located for a large number of design types. The disadvantage can be blurring distinctive characteristics between design types and as such it will take longer for the process to learn precisely to which design type it belongs.

Re-scaling can be achieved through stretching or compressing a design to the unit cube, so that the length of each dimension is re-scaled such that 0 is the minimum and 1 the maximum in that dimension for that design. If such a transformation is performed, care must be taken in interpreting distance between two points as the scales would not be the same between dimensions. Rotations can be the result of a non-standardized axis used with designs. As such, through rotation the data we are constructing a

common axis, which may reveal more similarity across the designs. Rotation data is achieved through matrix multiplication of the data set. For example, in a 2 dimensional design, every rotation around the origin in a counter clockwise direction can be represented with the matrix $R$ shown in Eq. (15).

$$R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{15}$$

When the data are multiplied by $R$ we obtain the new coordinates as in Eqs. (16).

$$\begin{aligned} x_j(\theta) &= \cos(\theta)x_j - \sin(\theta)y_j \\ y_j(\theta) &= \sin(\theta)x_j + \cos(\theta)y_j \end{aligned} \tag{16}$$

An analyst could decide upon rotation and rescaling based on visual inspection. However, for a more rigorous approach we would need to measure the distance between designs and seek to minimize it. Following the approach proposed by [31] we use the Kullback-Leibler (KL) divergence measure to assess the difference between designs. Using the superimposition of all designs as an average design we can measure the difference of each design to the average and seek to minimize it.

The KL divergence measure of design type $v$ to $u$, denoted by $D_{KL}(D_u \parallel D_v)$ is given in Eq. (17).

$$D_{KL}(D_u \parallel D_v) = \sum_{m=1}^{m_{max}} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} P_{u,m} \left( \frac{\sum_{j=1}^{n_{u,m}} \phi_j(x, y; \mu_{u,x,j} = x_{u,m,j}, \mu_{u,y,j} = y_{u,m,j}, \sigma)}{n_{u,m}} \right) \ln \left( \frac{P_{u,m} \left( \frac{\sum_{j=1}^{n_{u,m}} \phi_j(x, y; \mu_{u,x,j} = x_{u,m,j}, \mu_{u,y,j} = y_{u,m,j}, \sigma)}{n_{u,m}} \right)}{P_{v,m} \left( \frac{\sum_{j=1}^{n_{v,m}} \phi_j(x, y; \mu_{v,x,j} = x_{v,m,j}, \mu_{v,y,j} = y_{v,m,j}, \sigma)}{n_{v,m}} \right)} \right) \mathrm{d}x\mathrm{d}y \tag{17}$$

This is re-expressed in Eq. (18).

$$D_{KL}(D_u \parallel D_v) = \mathbb{E}\left[\ln\left(p_{u,m}\left(\frac{\sum_{j=1}^{n_{u,m}} \phi_j(x, y; \mu_{u,m,j} = x_{u,m,j}, \mu_{u,y,j} = y_{u,m,j}, \sigma)}{n_{u,m}}\right)\right)\right] - \mathbb{E}\left[\ln\left(p_{v,m}\left(\frac{\sum_{j=1}^{n_{v,m}} \phi_j(x, y; \mu_{v,m,j} = x_{v,m,j}, \mu_{v,y,j} = y_{v,m,j}, \sigma)}{n_{v,m}}\right)\right)\right] \tag{18}$$

Where the expectation is taken with respect to the distribution with $u$. Expressing this as an expectation provides a computational advantage, as a closed form analytical solution is not available, we can conduct Monte Carlo simulations with the distribution of $u$ and evaluate the average of the expression. In sum, we can transform each design through rotation and re-scaling to minimize the KL divergence of the mean design to the design in question.

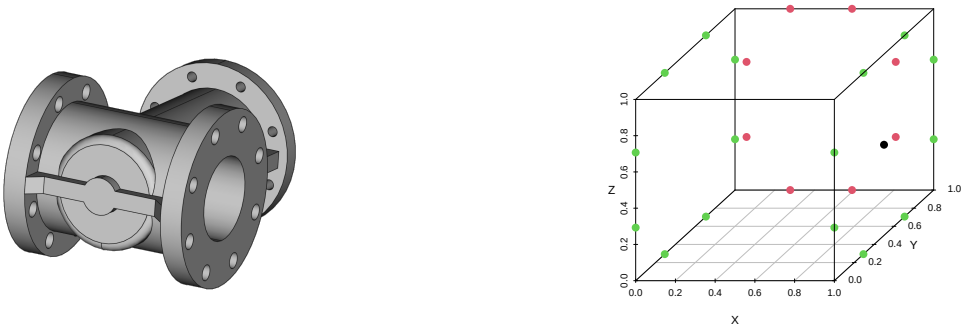
### 4.7 Summary

Section 4 has outlined the underlying model and process to support the prediction of features given location. This can be used with an interactive CAD system, where the cursor sits in a location described by its coordinates and the recommended feature is suggested. In Section 5 we apply this to a data set.

## 5 Case Study

To allow an intuitive, visual understanding of the proposed process we have chosen to use a set of 513 mechanical valve designs. The structure of the valve bodies have obvious regularities with circles around the valve's flanges together with other functional holes. An unordered set of hole diameters and associated $(x, y, z)$ coordinates were extracted for each valve body from the B-rep of the CAD design using the Twig match algorithm [34]. Further details are provided in [31]. An example valve design is shown in Fig. 4a with the extracted hole features, scaled to $[0, 1]$, shown in Fig. 4b.

In this analysis, the aim is to predict the sequential addition of hole features and their position given the state of the current design, with the focus on features occurring on the same surface plane i.e. predicting a hole diameter on the flange surface.




(a) (b)

Figure 4: Example of a valve body from CAD design database. Figure 4a shows an image of a valve design and Fig. 4b the scaled positions of the extracted hole features. The different colors are used to identify the different diameters of the holes

## 5.1 Scaling

To facilitate prediction the feature coordinates of each design were scaled to the unit cube – each dimension was scaled to $[0, 1]$ – and additional rescaling was required on each cube surface so that the features retained their geometric shape. Models were then estimated using the features and feature positions which were positioned on the surface of the cube, one surface at a time. For example, after scaling, the data were subset to analyze the features on the $x = 0$ face. Figure 5 shows the superimposition of the scaled feature coordinates from all designs on the $x = 0$, $y = 0$ and $z = 0$ faces (some jitter of the points has been added to the Figure to aid feature discrimination).

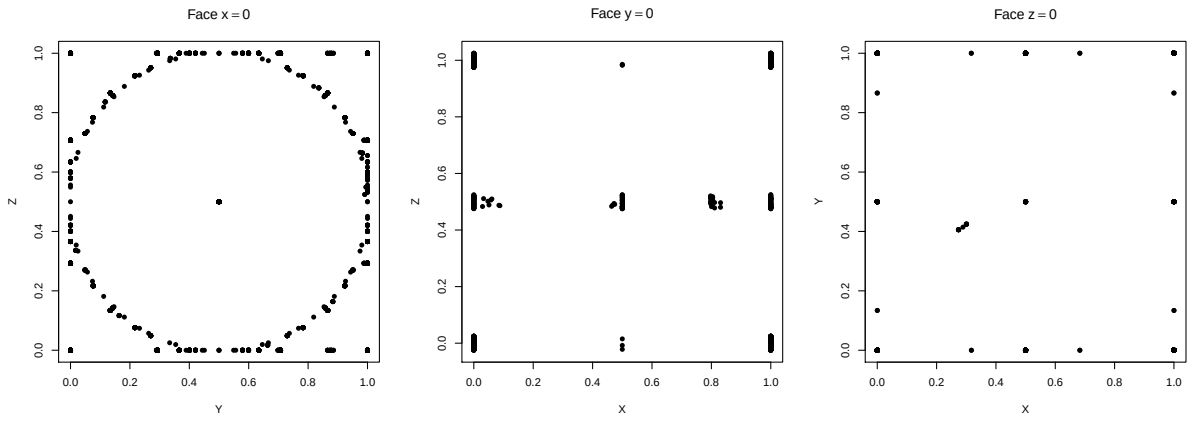



Figure 5: Superimposition of scaled hole feature coordinates for all the data on the specified face of the cube.

## 5.2 Kernel Density Estimation

The KDE were estimated across each face of the cube. This was done by dividing the face into $N$ by $N$ regular grid positions and then estimating the kernel density at each grid position for each feature in all designs. A Normal kernel with user specified standard deviation was applied, in this analysis chosen to be 0.05, and the density on each dimension calculated independently. If a design did not have a specific feature that was present in the database pool of features then a Uniform probability across the grid of positions was assumed. This allows for predictions to be generated on a new design which is using a combination of hole features that have not been previously observed. The KDE outputs a density estimation at each position in the $N$ by $N$ grid for every feature in the database.

## 5.3 Evaluation

Both the correctness of the FLPF and the predictive accuracy of the PFLF were assessed using 10-fold cross-validation. The kernel density across the features was estimated using the training data and then evaluated on the designs in the test set. Each test design, which contain hole feature labels and their coordinates, was evaluated one at a time. Three measures of predictive performance were used; the distance from the observed feature coordinate to the nearest predicted mode was calculated using two approaches, and reciprocal rank was used to evaluate how accurately a feature was predicted at its observed position. Further details are provided in the Supplemental Material S3.3.

An illustration of the Bayesian updating and predictions on one test design is provided in the Supplemental Material S3.

## 5.4 Results

Figure 6 illustrates the aggregated results from the cross-validation. The x-axis indicates how many holes have been added to a new design (e.g. if a test design has four features, then the predictive densities, distances to mode and ranks are calculated after sequentially adding 0, 1, 2, or 3 holes to the new design). The y-axis gives the mean distance to the mode, either on the raw scale or in grid steps or the mean reciprocal rank. The red triangle gives the mean across the ten folds. The performance of the predicted rank of suggestions is shown in the third figure; the range of values is from zero – poor suggestions, to one – perfect suggestions.

As expected an initial improvement is observed in the distances to the nearest predictive mode as additional features were added to each new test design, however, there is a clear pattern of extreme values within all figures which results in a decrease in performance as additional holes are added to a design. This can be explained, as within each test fold there are a few designs which are unlike anything in the training set and thus the KDE does not provide reasonable predictions.

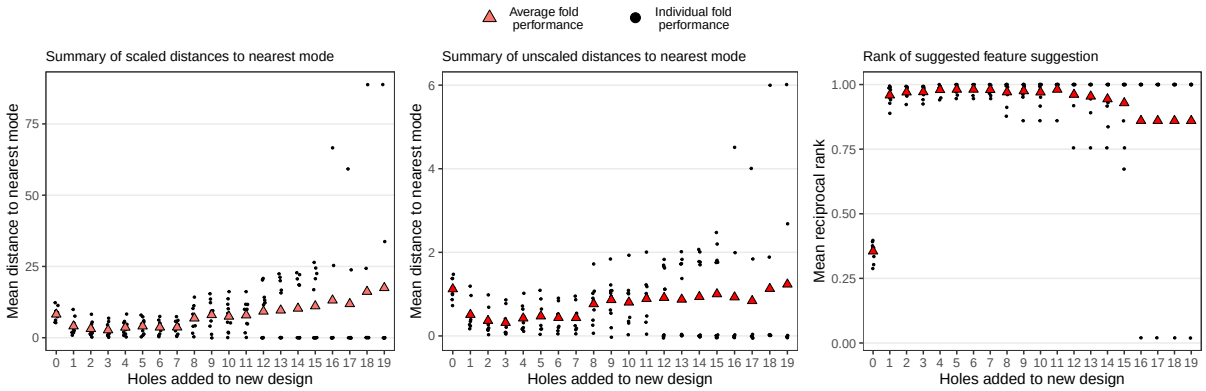



Figure 6: Cross-validation predictive performance. The x-axis indicates how many holes have been added to a new design and the y-axis records the average distance or rank. The black points give the within fold average measure and the red triangle the average performance across the folds.

Some examples follow. All the designs in which a specific feature occurs in the training dataset may have a different number of feature instances than observed in the test design. In one example, within a training dataset instance, designs with the hole diameter "33.0" have between 3 and 13 instances on the flange plane, however it occurs 16 times within a test design. This results in all the predictive modes being slightly offset, as seen in the Supplemental Material S6.1. This modeling framework cannot infer the coordinates for features even though they are still placed within the same circular orientation. Another example is that all of the designs with a specific feature in the training dataset are positioned differently than those in the test design. For an example shown in the Supplemental Material S6.2, the hole diameter "35.0" was used as a central bore hole in the training data designs, however, it was used as the bolt connector within the new test design. Therefore as additional holes are added to the new design the updated predictive density provides little information. Clearly the order in which the features are added to a new design will affect the predictive density, particularly when there are multiple types (hole dimensions) of feature, and this can impact the quality of predictive guidance. For an example shown in the Supplemental Material S6.3, there are 30 designs within a training dataset instance that contain the "22.0" diameter, but only one of these also has the additional "17.29" diameter. The early selection of the "17.29" feature adds more probability weight onto the single design in the training set, and it takes several further additions for the predictions to improve.

The predictive performance of the method was re-evaluated omitting the 24 unusual designs from the test datasets and the results are shown in Fig. 7. This is done to examine the predictive performance of the model for a designer who remains within the catalog of previous designs. It can be seen that the predictive performance improves as more features are added to a new design. This again indicates that the utility of the method is dependent on the designs forming a homogeneous set. The folds with larger values can be explained by the ordering of the features entering in to the design, as illustrated by the example given in the Supplemental Material S6.3. While the distance to the nearest predictive mode

may be small, there remain extreme values in the rank predictions. This indicates that while a feature is expected at a position, our model has been unable to predict the specific feature, and so suggests that the feature added at this position is unusual given those observed at in the training data.

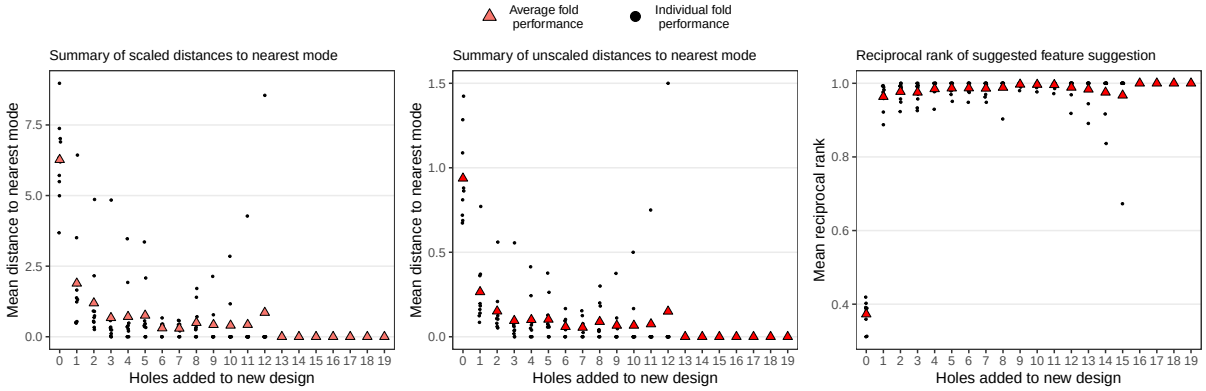



Figure 7: Cross-validation predictive performance after omitting unusual designs from the test set. The x-axis indicates how many holes have been added to a new design and the y-axis records the average distance or rank. The black points give the within fold average measure and the red triangle the average performance across the folds.

The prediction results for the features on the $y = 0$ and $z = 0$ cube face are provided in the Supplemental Material S7. Performance is similar to the $x = 0$ face with predictions improving as additional features are added to the design. Within the features on these faces, there are two designs that are unlike any of the other designs; the effect of this is more apparent on the $y = 0$ face. Again removal of the outlier design resulted in improved statistics (results not provided).

## 5.5 Association Measure

The utility of the method is supported by the homogeneity of the design database. Section 4.5 described how the mutual information could be used to provide some measure of the expected dependence in a database, however, there is no analytic solution to Eq. (14) for our non-parametric model. We therefore estimate this measure through a simulation exercise. We denote $i_{max}$ as the number of designs in the database, $m_{max}$ as the number of unique marks (features) and $n_{i,m}$ as the number of marks of type $m$ in design $i$. The probability of randomly choosing design $i$ is given by $q_i = {}^1/_{i_{max}}$ and the probability of selecting mark $m$ given that design $i$ was selected is defined by $p_{i,m} = {}^{n_{i,m}}/\sum_{\forall m} n_{i,m}$.

For a given design database, a representative random sample of feature instances is generated, using the following steps,

1. Uniformly sample a design $i$ from the set of designs in the database with probability $q_i$
2. Randomly sample a feature type $m$ from design $i$ with probability $p_{i,m}$
3. Sample a single instance of feature type $m$, as there may be multiple instances of feature $m$ within design $i$.

4. Take a random sample from the Normal kernel with mean at the $(x, y)$ coordinates

5. Repeat many times

An estimate of the MI can then be calculated using the expressions given in the Supplemental Material S2 where the KDE of the designs in the database are evaluated at the sampled coordinates.

The dependence structure within our sample database was estimated using this method across the ten training cross-validation datasets, and the resulting MI scores had mean 1.43 and standard deviation 0.03. This equates to a scaled $\dot{\omega} = 0.97$ indicating that there is strong dependence within the data and thus we would expect predictions to be good. For comparison, a null distribution was estimated for the statistic on the same training data, permuting the feature instance and generating the coordinates randomly from the Uniform [0,1] distribution, so that features were no longer aligned with specific designs or coordinates. This gave a MI of 0.62 (0.03), and scaled value of 0.84. A second smaller simulation of randomly generated designs and feature coordinates revealed that smaller samples produced higher MI. As sample size increased then the MI decreased to zero, the theoretical value for independence. It would therefore be useful for practitioners to evaluate the MI on randomly permuted data to support interpretation of the MI score on the design database.

## 5.6 Rotation

The more similar the designs in the database the stronger the signal for making predictions. However, different designs may have been created with a different orientation. Section 4.6 described how the KL measure could be used to rotate one design to minimize the probabilistic differences between them. There is not an analytic solution to Eq. (18) for our non-parametric model but we can minimize the KL divergence between two designs $u$ and $v$, $D_{KL}(D_u \parallel D_v)$, by finding the angles of rotation that maximize the second term by a simulation design embedded in an optimization routine. As the KDE may be a noisy function, a global optimization routine should be used, although a brute force search is feasible in 2D.

The rotation can be implemented as follows. The design database contains feature instances that are assumed to be representative of the underlying orientation of the designs and we consider this as a single average design. Each design would then be orientated in turn, to this average design, excluding the design getting rotated from the pool. First multiple random draws are simulated from the Normal kernel, with means equal to the feature positions of the design to be rotated. Then the KL measure is calculated between the samples and the average design. The new design is then rotated and re-sampled until the KL measure is minimized. An illustration of the 2D rotation of a part is given in the Supplemental Material S5.

## 5.7 Choice of Standard Deviation

The choice of standard deviation (bandwidth) of the Normal kernel determines how spread out the predicted density is around the training data observations. Figure 8 shows the predicted density for the same training data under different kernel standard deviations. There are two main approaches one could take in determining a suitable value for this parameter. It may be that prediction is the only desirable performance measure and then through cross validation exercises an optimal value can be identified. Alternatively, with more emphasis on the prospective nature of this decision support, to facilitate the determination of designs that are similar to historical designs, this parameter can be used as a controlling lever, whereby small values will result in predictions that are very close to previous designs.

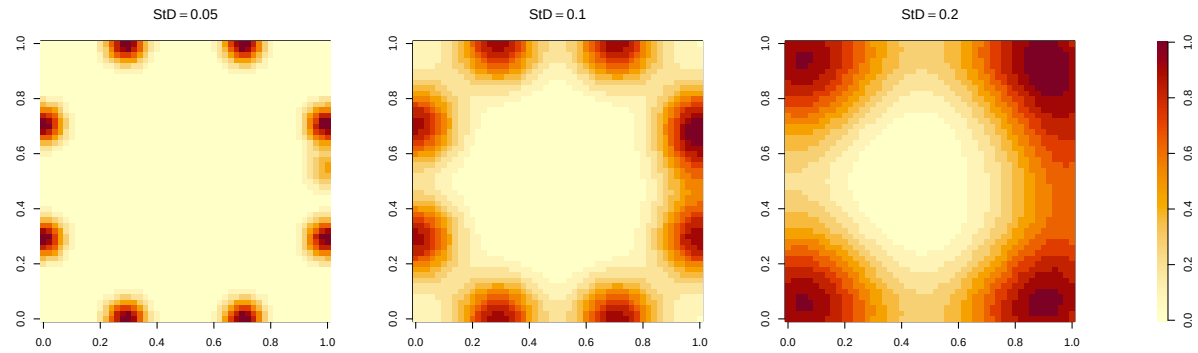



Figure 8: Predictive density calculated under a Normal kernel using different standard deviations. The colourbar legend describes how the image colour maps to the data, with dark red indicating regions of higher predictive density

## 5.8 Discussion of Results

We have described how to implement the proposed process for predicting the type and location of the features that might be added during an engineering design process. We evaluated the method on a data set of real designs through a cross validation process. In 90% of the evaluation runs the feature's actual location and the prediction (once at least one feature had been selected) were very close (i.e. within 0.5 grid space on average – 1% of the normalized range of the part). When more features were added to the design the accuracy of the predictions improved. This observation can be clearly seen in the ranking of the predicted features (i.e. an ordered list of the most to least likely features to occur at a given location). If four features had been selected (i.e. added to the design) the subsequent features selected were, on average, ranked in the first 25% of the list of suggestions. This increased to the top 10% once eight features had been selected.

This behavior reflects the nature of the commercial product families which formed the dataset. These have frequently repeated sets of features at standardized positions within a design, and so after one choice has been made then subsequent choices can be predicted with a high probability. In other words portfolios

of mechanical designs have strong dependence in data that results in strong predictive performance.

Figure 9 shows two components with a feature prediction "heat map" manually superimposed on to their faces. These hotspots indicate where features (regardless of their type and size) are frequently located in the training data. For each figure, the left-hand side plot presents the complete part, the central plot presents the predictive density from the training pool when no features have been added to the part, and the right-hand side plot the updated density given that the new features have been added to the design.

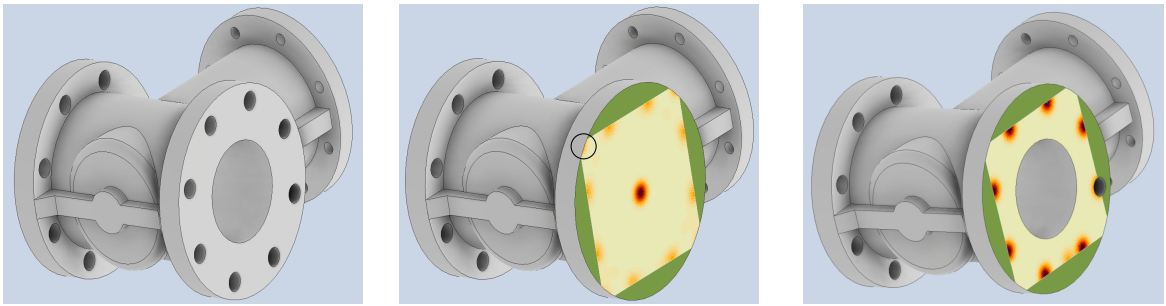
Figure 9b presents an example of how the predictive density changes once a feature has been added to the design. Before a choice has been made, the greatest predictive density is placed on the four corners of the feasible box, the normalized range of the scaled prediction region, indicating that this orientation is most common in the training dataset. Once a feature of specific size and position has been added, the predictive density changes to favor a pattern of six holes away from the corners. Red and blue circles are used in Fig. 9b to highlight how the predictive density in these regions change given some feature addition. Although such heat-maps give an intuitive overview the prediction results can be presented to a user in several different ways. For example, given a location (e.g. the user's curser) a list of feature suggestions (ranked in order of their likelihood) could be generated.

The heat maps also illustrate the need for further research into user interfaces that allow the designer to control the choice of training data (used to generate the predictions) and the scaling/mapping of the results onto new designs. In the case study, the feature coordinates were normalized to boundaries determined by the extent of feature locations within the training dataset. For example, in Fig. 9a predictions were generated across the unit cube and so there is non-zero density in the corners (highlighted by black circle in central panel of Fig. 9a), whereas if a unit circle had been used to normalize the feature locations the result would be more appropriate to the shape by omitting the truncated predicted region corners. An obvious artifact of the current approach to normalization is that there will be regions on a face (colored green in Fig. 9) that are beyond the geometric extent of the features used to train the prediction system. Due to the restrictions that we have imposed through this mapping, predictions were not generated outside the range of the normalized region. However, depending on the choice of kernel, one could extrapolate beyond these boundaries for a new design, but as with any extrapolation, these require stronger assumptions.

This could be mitigated by filtering suggestions that are physical or functionally feasible before presenting them as options to the designer. The development of effective filters would also enable the geometric limits on feature prediction to be determined in a manner most appropriate to the application (e.g. a part bounding box, or specific planes). However, work is required on generic scaling functions to support this, for example the top flange of the design-part provided in Fig. 9b is rectangular and so the predictive density requires to be mapped back to the original part dimension from the unit cube used to

generate predictions.

(a) Part-design A



(b) Part-design B

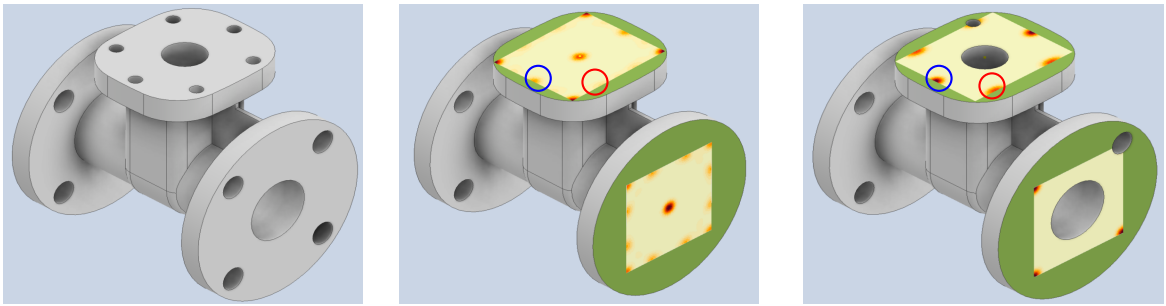


Figure 9: The feature predictions were mapped onto two incomplete designs. Regions of higher predictive density are illustrated by dark red. From left to right, the plots show the complete part, the predictive distribution when no features have been added, and so is completely informed by the training data, and lastly the predictive distribution after the addition of a central bore hole and one bolt hole feature. The updated distribution displays the increased level of belief in the positions of subsequent features, with areas of higher density and lower variance. The black circle in (a) is used to indicate the prediction region which has been truncated and the red and blue circles in (b) are used to highlight the changes in the predictive density once a choice has been made. The green sections represent areas outside the normalized prediction region.

# 6 Summary and Conclusions

The aim of this research was to "Define a computational framework that can support an interactive design process with suggestions of features based on three inputs: a knowledge of existing designs; the state of an emerging design and a location on the surface of the emerging design." The authors believe that system described meets this goal and has established how the feature content of mechanical designs can be amalgamated and transformed into a likelihood function that defines the probability of particular design features occurring at specific locations on a model.

The work has not only demonstrated that the architecture of the proposed system is viable but also established that the computations can be done quickly enough to support a dynamic design process. For example, the prototype system can respond to a given mouse location at interactive speeds (i.e. ms) and consequently could support user interface functionality such as pop-up menus (customized to reflect

likely feature types and parameter values) or even ghosting images of possible features onto the cursor location as it moves to particular locations. In this way the engineer is free to ignore these selections in the same way a user of a predictive text system is able to adopt or dismiss suggestions when composing SMS texts.

The case-study illustrated the method using hole features, however, the feature set could be extended to include those with a more complicated geometry. Provided that a feature can be defined geometrically and hence extracted from the CAD design, the prediction method can be applied by considering such features as another type of mark. This would allow for modeling the dependence both between and across feature types.

## 6.1 Limitations

Like other predictive systems and there are inevitable limitations. Currently the system can only predict the likelihood of features occurring within the volume defined by the maximum extent of the features extracted from the training dataset. Understanding how these results can be generalized to support predictions across variable volumes, as well as optimal scaling of the normalized prediction region, is an area of further research. Additionally, while the method of data homogenization appears to be viable for product families with very regular structures (e.g. industrial valves or manifold blocks), its behavior with product families with more variable forms is not clear.

However, one of the features of all interactive predictive systems (that makes them viable) is that the user is always free to ignore suggestions that are wrong or out of context. In other words predictive systems do not have to provide perfect predictions all the time to be useful.

## 6.2 Future Work

Having established the fundamentals of the theory the authors intend to broaden the application to other datasets of mechanical component designs. This will allow the investigation of the methodology's ability to support multiple feature types and more geometrically varied product families i.e. the scaling of the normalized prediction region. The merits and implications of estimating the normalized prediction region using different kernels which can account for boundary effects will be studied. Considering MPP's beyond simple Euclidean geometry provide opportunities. The current focus of the project has been on providing decision support to a single engineer, and how such a system will support concurrent designs carried out simultaneously by distributed teams is a topic that requires further investigation.

Although this work has established the theoretical and computational foundations for a predictive system its utility will ultimately depend on how its user interface behaves. Although beyond the scope of this work follow-on projects will seek to incorporate the predictive functionality described in a commercial CAD system (via their API) and so allow a systematic assessment of the impact of predictive CAD on

design productivity to be undertaken.

## 6.3 Acknowledgements

This work was supported by the Engineering and Physical Sciences Research Council, UK [grant number EP/R004226/1]. The dataset of hole features extracted from the valves models is available at `https://doi.org/10.15129/c3ae80a3-b34e-43cc-b6e6-629eb5cab922` (the URL will become active on journal acceptance).

# References


[1] John E Ettlie and Matthew Kubarek. Design reuse in manufacturing and services. *Journal of Product Innovation Management*, 25(5):457–472, 2008.

[2] Rob Bracewell, Ken Wallace, Michael Moss, and David Knott. Capturing design rationale. *Computer-Aided Design*, 41(3):173–186, 2009.

[3] Christopher McComb, Jonathan Cagan, and Kenneth Kotovsky. Mining process heuristics from designer action data via hidden markov models. *Journal of Mechanical Design*, 139(11), 2017.

[4] Ayush Raina, Christopher McComb, and Jonathan Cagan. Learning to design from humans: Imitating human designers through deep learning. *Journal of Mechanical Design*, 141(11), 2019.

[5] Suyu Hou and Karthik Ramani. Dynamic query interface for 3d shape search. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 46970, pages 347–355, 2004.

[6] Silvia Schacht and Alexander Mädche. How to prevent reinventing the wheel?–design principles for project knowledge management systems. In *International Conference on Design Science Research in Information Systems*, pages 1–17. Springer, 2013.

[7] John Ashburner and Karl J Friston. Voxel-based morphometry—the methods. *Neuroimage*, 11(6): 805–821, 2000.

[8] A. Getis, J. Lacambra, and H. Zoller, editors. *Spatial econometrics and spatial statistics*. Citeseer, 2004.

[9] Siddhartha Chaudhuri and Vladlen Koltun. Data-driven suggestions for creativity support in 3D modeling. *ACM Transactions on Graphics*, 29(6), 2010.

[10] Siddhartha Chaudhuri, Evangelos Kalogerakis, Leonidas Guibas, and Vladlen Koltun. Probabilistic reasoning for assembly-based 3D modeling. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 30(4), 2011.

[11] Evangelos Kalogerakis, Siddhartha Chaudhuri, Daphne Koller, and Vladlen Koltun. A probabilistic model for component-based shape synthesis. *ACM Transactions on Graphics (TOG)*, 31(4):1–11, 2012.

[12] Prakhar Jaiswal, Jinmiao Huang, and Rahul Rai. Assembly-based conceptual 3D modeling with unlabeled components using probabilistic factor graph. *Computer-Aided Design*, 74:45–54, 2016.

[13] Laureen Lam, Sharon Lin, and Pat Hanrahan. Using text n-grams for model suggestions in 3D scenes. In *SIGGRAPH Asia 2012 Technical Briefs*, pages 1–4. 2012.

[14] Matthew Fisher, Manolis Savva, and Pat Hanrahan. Characterizing structural relationships in scenes using graph kernels. *ACM Transactions on Graphics*, 30(4), 2011.

[15] Siddhartha Chaudhuri, Evangelos Kalogerakis, Stephen Giguere, and Thomas Funkhouser. Attribit: content creation with semantic attributes. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*, pages 193–202, 2013.

[16] Adriana Schulz, Ariel Shamir, David IW Levin, Pitchaya Sitthi-Amorn, and Wojciech Matusik. Design and fabrication by example. *ACM Transactions on Graphics (TOG)*, 33(4):1–11, 2014.

[17] Tianqiang Liu, Siddhartha Chaudhuri, Vladimir G Kim, Qixing Huang, Niloy J Mitra, and Thomas Funkhouser. Creating consistent scene graphs using a probabilistic grammar. *ACM Transactions on Graphics (TOG)*, 33(6):1–12, 2014.

[18] Minhyuk Sung, Hao Su, Vladimir G Kim, Siddhartha Chaudhuri, and Leonidas Guibas. Complementme: weakly-supervised component suggestions for 3D modeling. *ACM Transactions on Graphics (TOG)*, 36(6):1–12, 2017.

[19] AJ Baddeley and MNM Van Lieshout. Stochastic geometry models in high-level vision. *Journal of Applied Statistics*, 20(5-6):231–256, 1993.

[20] Xavier Descombes. Multiple objects detection in biological images using a marked point process framework. *Methods*, 115:2–8, 2017.

[21] Xavier Descombes, Robert Minlos, and Elena Zhizhina. Object extraction using a stochastic birth-and-death dynamics in continuum. *Journal of Mathematical Imaging and Vision*, 33(3):347–359, 2009.

[22] Mathias Ortner, Xavier Descombes, and Josiane Zerubia. Building outline extraction from digital elevation models using marked point processes. *International Journal of Computer Vision*, 72(2): 107–132, 2007.

[23] Akos Utasi and Csaba Benedek. A 3-d marked point process model for multi-view people detection. In *CVPR 2011*, pages 3385–3392. IEEE, 2011.

[24] Maria S Kulikova, Ian H Jermyn, Xavier Descombes, Elena Zhizhina, and Josiane Zerubia. Extraction of arbitrarily-shaped objects using stochastic multiple birth-and-death dynamics and active contours. In *Computational Imaging VIII*, volume 7533, page 753306. International Society for Optics and Photonics, 2010.

[25] Dae Woo Kim, Camilo Aguilar, Huixi Zhao, and Mary L Comer. Narrow gap detection in microscope images using marked point process modeling. *IEEE Transactions on Image Processing*, 28(10):5064–5076, 2019.

[26] Quanhua Zhao, You Wu, Huabin Wang, and Yu Li. Road extraction from remote sensing image based on marked point process with a structure mark library. *International Journal of Remote Sensing*, 41(16):6183–6208, 2020.

[27] Amal Mbarki and Mohamed Naouai. A marked point process model for visual perceptual groups extraction. In *2020 IEEE International Conference on Visual Communications and Image Processing (VCIP)*, pages 511–514. IEEE, 2020.

[28] Xavier Descombes. *Stochastic geometry for image analysis*. John Wiley & Sons, 2013.

[29] John Quigley and Lesley Walls. A methodology for constructing subjective probability distributions with data. In *Elicitation*, pages 141–170. Springer, 2018.

[30] John Quigley, Gavin Hardman, Tim Bedford, and Lesley Walls. Merging expert and empirical data for rare event frequency estimation: Pool homogenisation for empirical bayes models. *Reliability Engineering & System Safety*, 96(6):687–695, 2011.

[31] Gokula Vasantha, David Purves, John Quigley, Jonathan Corney, Andrew Sherlock, and Geevin Randika. Common design structures and substitutable feature discovery in CAD databases. *Advanced Engineering Informatics*, 48:101261, 2021. ISSN 1474-0346. doi: https://doi.org/10.1016/j.aei.2021.101261. URL `https://www.sciencedirect.com/science/article/pii/S1474034621000161`.

[32] John Quigley and Matthew Revie. Estimating the probability of rare events: addressing zero failure data. *Risk Analysis: An International Journal*, 31(7):1120–1132, 2011.

[33] Harry Joe. Relative entropy measures of multivariate dependence. *Journal of the American Statistical Association*, 84(405):157–164, 1989.

[34] Duncan Paterson and Johnathan Corney. Feature based search of 3d databases. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 50084, page V01BT02A010. American Society of Mechanical Engineers, 2016.

[35] Stuart Russell and Peter Norvig. Artificial intelligence: a modern approach. 2020.

# Supplemental Material

## S1 Moments from KDE

$$\mathbb{E}[X] = \sum_{i=1}^{i_{max}} \sum_{m=1}^{m_{max}} \pi(i) p_{i,m} \left\{ \delta_{n_{im}} \frac{\sum_{j=1}^{n_{i,m}} x_{i,m,j}}{n_{i,m}} + (1 - \delta_{n_{im}}) \frac{1}{2} \right\} \tag{19}$$

$$\mathbb{E}[Y] = \sum_{i=1}^{i_{max}} \sum_{m=1}^{m_{max}} \pi(i) p_{i,m} \left\{ \delta_{n_{im}} \frac{\sum_{j=1}^{n_{i,m}} y_{i,m,j}}{n_{i,m}} + (1 - \delta_{n_{im}}) \frac{1}{2} \right\} \tag{20}$$

$$\mathbb{E}[XY] = \sum_{i=1}^{i_{max}} \sum_{m=1}^{m_{max}} \pi(i) p_{i,m} \left\{ \delta_{n_{im}} \frac{\sum_{j=1}^{n_{i,m}} x_{i,m,j} y_{i,m,j}}{n_{i,m}} + (1 - \delta_{n_{im}}) \frac{1}{4} \right\} \tag{21}$$

$$\mathbb{E}\left[X^2\right] = \sum_{i=1}^{i_{max}} \sum_{m=1}^{m_{max}} \pi(i) p_{i,m} \left\{ \delta_{n_{im}} \frac{\sum_{j=1}^{n_{i,m}} x_{i,m,j}^2}{n_{i,m}} + (1 - \delta_{n_{im}}) \frac{1}{4} \right\} \tag{22}$$

$$\mathbb{E}\left[Y^2\right] = \sum_{i=1}^{i_{max}} \sum_{m=1}^{m_{max}} \pi(i) p_{i,m} \left\{ \delta_{n_{im}} \frac{\sum_{j=1}^{n_{i,m}} y_{i,m,j}^2}{n_{i,m}} + (1 - \delta_{n_{im}}) \frac{1}{4} \right\} \tag{23}$$

## S2 Mutual Information Distributions

The Mutual Information measure for the model is defined by $\omega$ in the following.

$$\omega = \mathbb{E}\left[\ln\left(f\left(X,Y,M\right)\right)\right] - E\left[\ln\left(f\left(X\right)\right)\right] - E\left[\ln\left(f\left(Y\right)\right)\right] - E\left[\ln\left(f\left(M\right)\right)\right]$$

Therefore we require the entropy measures for the joint and marginal distributions.

As we are investigating dependency within the historic data we use the following joint and marginal probability density function where $\delta_{n_{im}}$ has been set to 1.

$$f\left(x,y,m\right) = \sum_{i=1}^{i_{\max}} \sum_{m=1}^{m_{\max}} \pi\left(i\right) \delta_m p_{i,m} \left( \frac{\sum_{j=1}^{n_{i,m}} \phi_j\left(x,y;\mu_{x,j}=x_{i,k,j},\mu_{y,j}=y_{i,m,j},\sigma\right)}{n_{i,m}} \right)$$

$$f\left(x\right) = \sum_{i=1}^{i_{\max}} \sum_{m=1}^{m_{\max}} \pi\left(i\right) p_{i,m} \frac{\sum_{j=1}^{n_{i,m}} g_j\left(x;\mu_{x,j}=x_{i,m,j},\sigma\right)}{n_{i,m}}$$

$$f\left(y\right) = \sum_{i=1}^{i_{\max}} \sum_{m=1}^{m_{\max}} \pi\left(i\right) p_{i,m} \frac{\sum_{j=1}^{n_{i,m}} g_j\left(y;\mu_{y,j}=y_{i,m,j},\sigma\right)}{n_{i,m}}$$

$$f\left(m\right) = \sum_{i=1}^{i_{\max}} \pi\left(i\right) p_{i,m}$$

Deriving the MI statistic will require either numerical integration of Monte Carlo simulations methods. We propose the latter.

Assume we generate $s'$ random simulations of $(x,y)$ locations and feature $m$ from $f(x,y,m)$. As we increase the number of simulations we can obtain a more accurate estimate of the expectation.

$$\mathbb{E}\left[\ln\left(f\left(X,Y,M\right)\right)\right] = \lim_{s'\to\infty} \frac{\sum_{s=1}^{s'} \ln\left(f\left(x_s,y_s,m_s\right)\right)}{s'}$$

Similar arguments hold for the marginal on location $(x,y)$. For feature direct calculation of the entropy is straightforward.

$$\mathbb{E}\left[\ln\left(f\left(M\right)\right)\right] = \sum_{i=1}^{i_{\max}} \sum_{m=1}^{m_{\max}} \pi\left(i\right) p_{i,m} \ln\left( \sum_{i=1}^{i_{\max}} \pi\left(i\right) p_{i,m} \right)$$

# S3 Case Study Demonstration

## S3.1 Feature Location Probability Function Assessment

For illustration, we step through an example using the data on the $x = 0$ face, and focus on one test design, and consider it a new design. With no features added to this on the $x = 0$ face, a uniform prior was assumed on which design from the training set the new design is most similar to, and use this to update the predictive distribution. Figure 10 provides a visualization of the predictive density. The darker red regions indicate areas of higher feature probability which reflects the contents on the training dataset with the central bore hole most prevalent. The blue numbers are used to indicate the positions of the hole features in the test design.

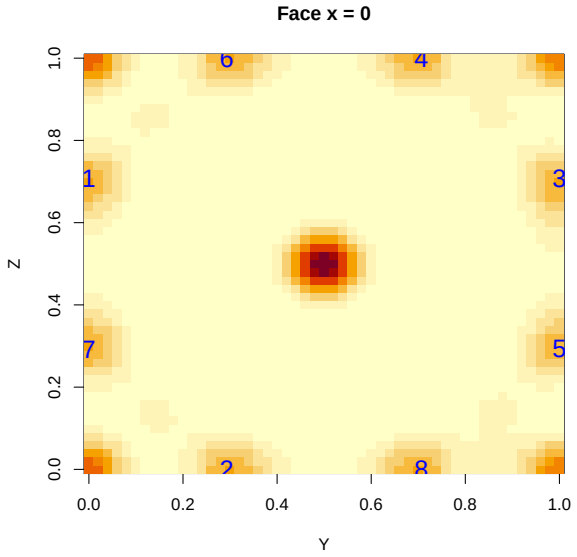



Figure 10: The predictive density for a new design given that no features have been added to the face with the blue numbers indicating the positions of the features that will be added showing strong agreement between the initial prediction and realization.

One feature is then added at the position observed in the test design. Given this additional information, the prior on which design from the training dataset the new design is most similar to is updated. Figure 11 illustrates how the prior on each training set design changes given this one observation; the red dashed line indicates the Uniform prior of no information and the probability of test designs either increase or decrease given the addition of the feature to the new design. Importantly, while the probabilities can be very small they are all non-zero. Moreover, we see an clear indication of a subset of designs types that are more likely candidates for the design being constructed.

## S3.2 Predictive Feature Location Function Predictive Performance

This posterior is then used to update the predictive distribution, with the density, aggregated over all features, shown in Fig. 12; the green triangle indicates the position of the feature that was added to the

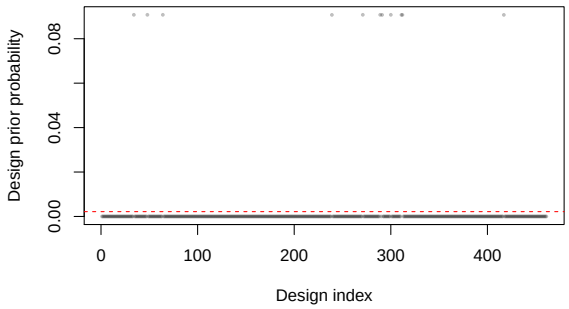



Figure 11: The posterior probability on which design from the training dataset a new design is most similar to. The red dashed line indicates the prior Uniform probability where all designs have equal probability.

new design. The addition of one feature shifts the probability to similar designs in the training dataset, and which captures the similar scaled coordinates.

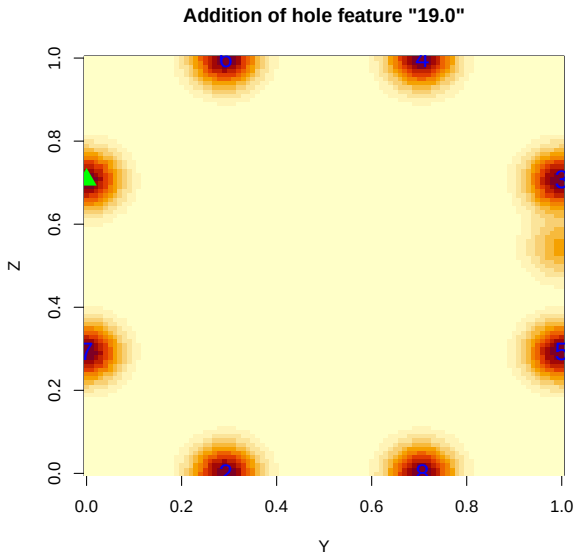



Figure 12: The updated predictive density for a new design given one feature, hole diameter "18.0", has been added at the position marked by the green triangle, showing stronger agreement between the prediction and the realizations than with the prior distribution.

Additionally, due to how the density is generated, the predictions can be presented at the feature level, and the ranked probability of feature inclusion calculated at each spatial coordinate. See Fig. 13 for the predictive distribution for the four features with the highest probability aggregated over the unit square; 94% of the predictive density weight is on the "19.0" feature, 5% on "6.4", and all other features having lower than 0.1% (note that the density color scale is not transferable between subplots i.e. the dark red may represent lower density between images).

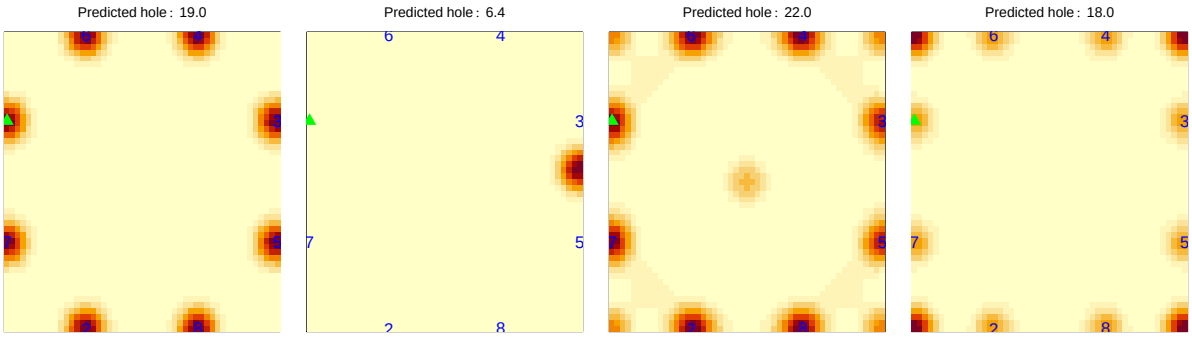



Figure 13: The updated predictive density for four different features for a new design given one feature, hole diameter "18.0", has been added at the position marked by the green triangle

### S3.3 Measuring Predictive Performance

Fig. 12 also illustrates how the system's predictive performance was evaluated. Using the test design, the observed features can be sequentially added to a new design at the observed test positions one at a time. A distance measure was then calculated after each addition between the positions of each of the remaining holes in the test design and the nearest predictive mode – the distance from the observed features in blue to the areas of high density in red – e.g. how well does the predictive density capture the feature positions. Details of how the distances were calculated are provided in S4.

A score was also generated to measure how accurately a hole feature is predicted at a specific position. The predicted density of each feature at the position of an observed hole in the test design were ranked, and this ranked list was then compared to the feature observed in the test design at that position; reciprocal rank was used as the performance measure.

These measures were calculated for all designs in the test datasets. The distances to remaining hole features and the ranks were averaged within each fold given how many holes have been added to the new design.

## S4 Evaluating Distance from Feature to Nearest Predictive Mode

A hill-climbing algorithm [35] was used to calculate the distance from the remaining features in the design to the nearest predictive mode; starting from the feature position moves in steps of one around the predicted intensity grid were evaluated until there were no increasing moves. The Euclidean distance between the feature position and the mode was then calculated. Euclidean distance was measured in two scales; one which accounts for the difference in scaling the dimensions to $[0, 1]$ and another in the raw counts of moving from one grid position to the next. For example in the Fig. 14, the predictive density on the $x = 0$ is calculated at discrete points in y and z, shown by the black points. We observe a point from the test design, in red. The search for the nearest predictive mode evaluates all of the adjacent grid positions to the current position in the search by comparing the density (dependent on scaled distance) of the neighboring discrete points to the observed; color blue. If a neighboring grid point has a higher value (green) then the search moves to that position. This continues until there are no increasing moves – this position is then taken as the nearest local mode to the observed point.

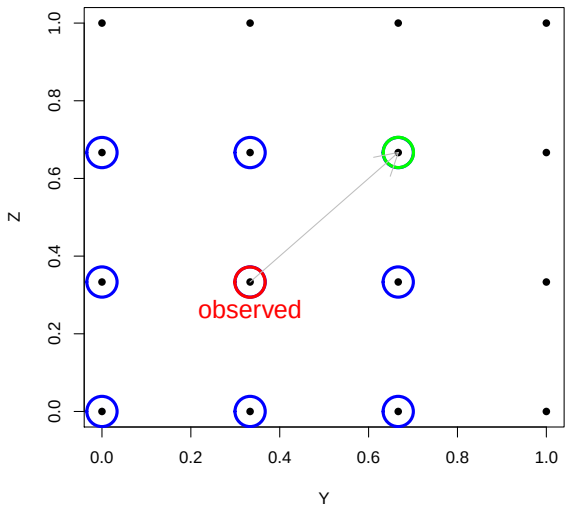



Figure 14: Illustration of hill-climbing search to nearest predictive mode. The $[0, 1]$ interval is divided into a $N$ by $N$ grid, in this case $N = 4$. The kernel density is estimated at the discrete black points. A feature is observed at a specific position, shown in red, and the mode is located by moving in increasing steps between neighboring grid positions.

## S5 Rotation of Features in Two Dimensions

The following example illustrates how the features of a part can be rotated to be more in alignment with the features in the database of designs. For an example, a part may have been designed around a rotated axis relative to the other parts in the database. Figure 15 shows the part from earlier rotated 20 degrees on the $x = 0$ face. The black points show the superimposition of all features instances from the training set on the $x = 0$ face – the average design, the blue triangles illustrate the positions of the features from a misaligned design – the design to be rotated – and the green stars give the feature positions post rotation. The rotation was seen to improve predictive ability; the average unscaled distance (grid points) from the features to the predicted modes was seven steps before rotation but one after.

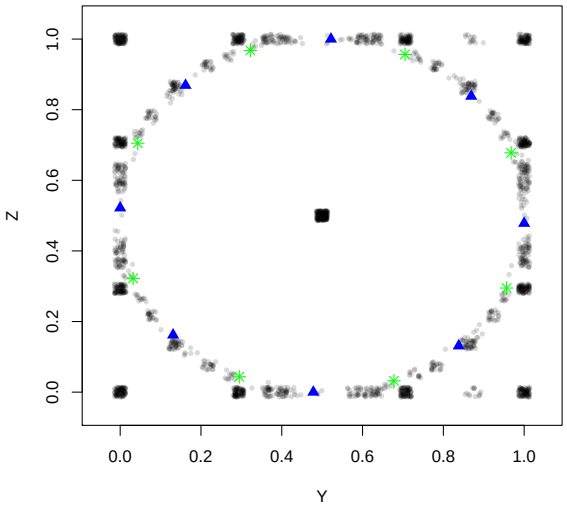



Figure 15: A superimposition of all features instances from the training set on the $x = 0$ face. The blue triangles illustrate the positions of the features from a misaligned design. The green stars give their positions post rotation

# S6 Illustration of Designs with Poor Predictive Performance

## S6.1 Different Spatial Orientation of Features

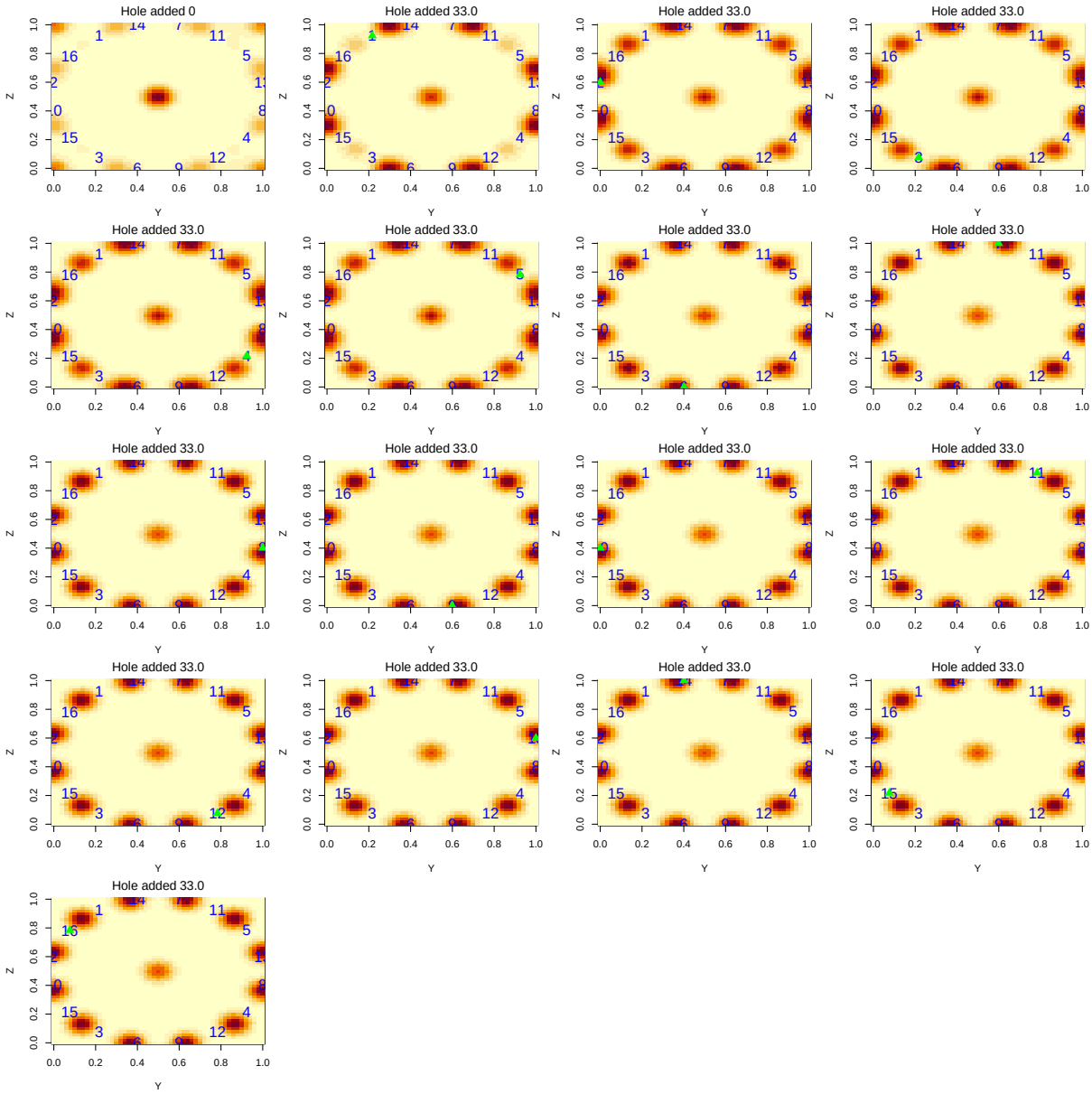



Figure 16: Predictive density for a new (test) design with greater feature occurrence than is present in the training data designs

## S6.2 Different Feature Use

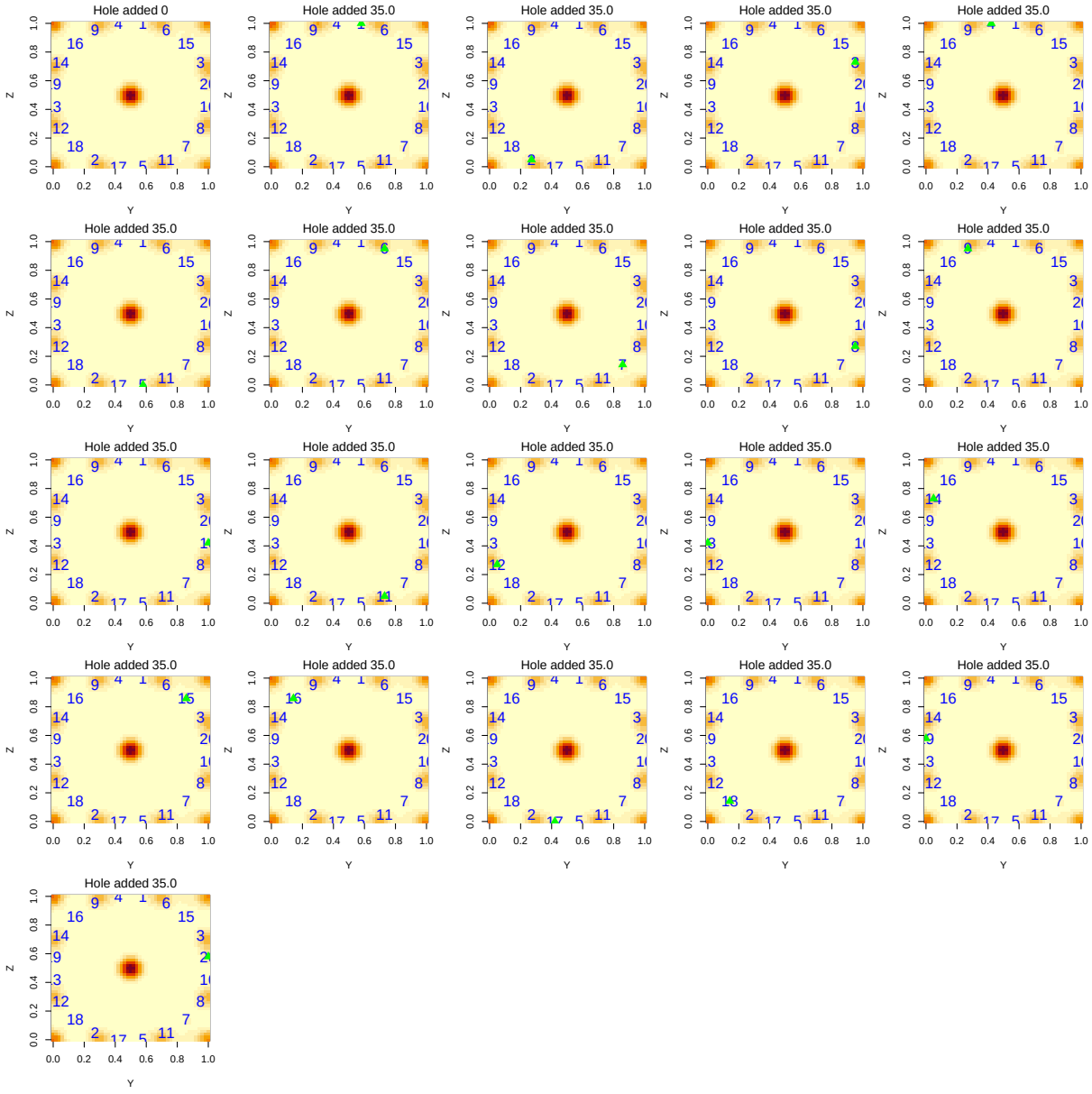



Figure 17: Predictive density for a new (test) design in which a hole feature is used for a different purpose than in the training data designs

## S6.3 Order of Sequential Feature Addition

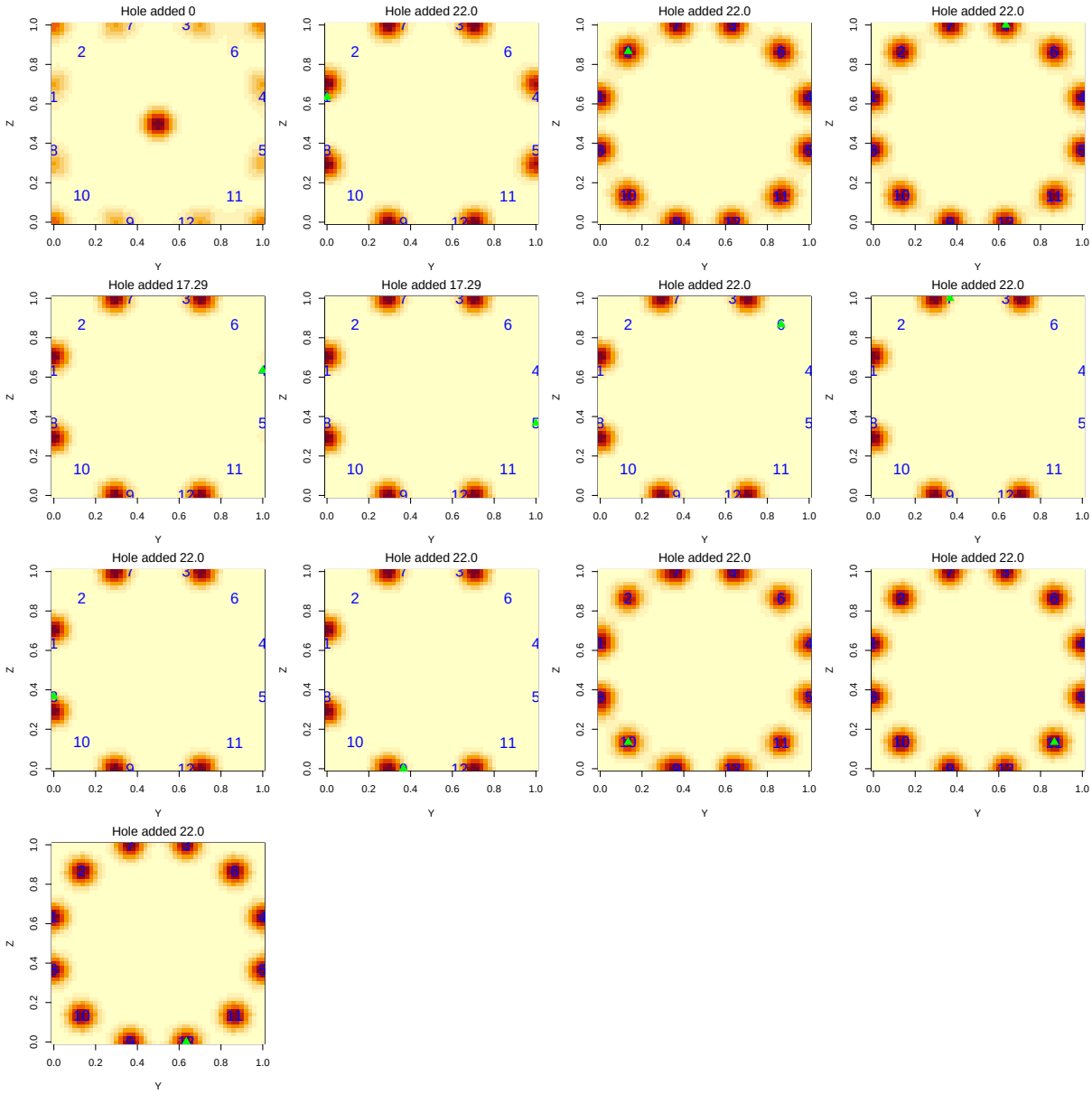



Figure 18: The order that hole features are added to a new design impacts the predictive density and thus decision support.

# S7 Additional Modeling Results

## S7.1 Prediction Results for the $y = 0$ Face

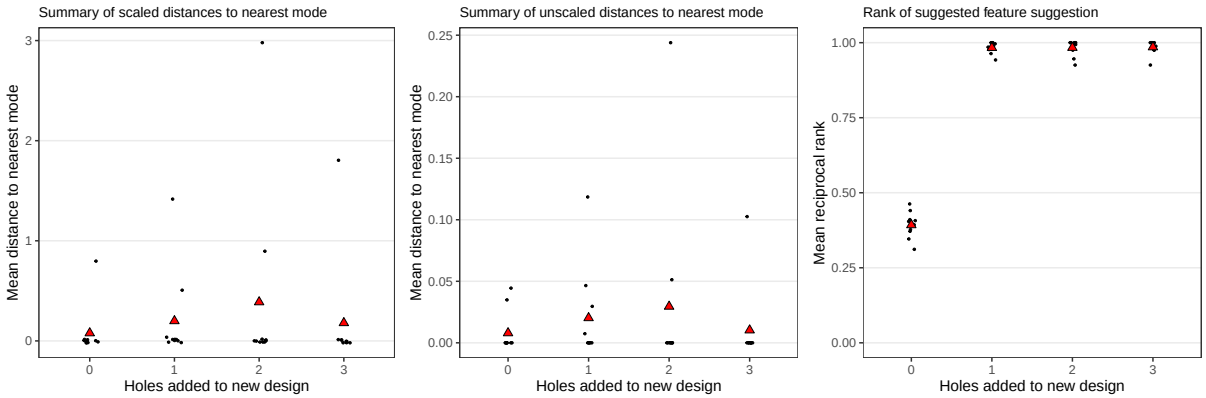



Figure 19: Cross-validation predictive performance. The x-axis indicates how many holes have been added to a new design and the y-axis records the average distance or rank. The black points give the within fold average measure and the red triangle the average performance across the folds.

## S7.2 Prediction Results for the $z = 0$ Face

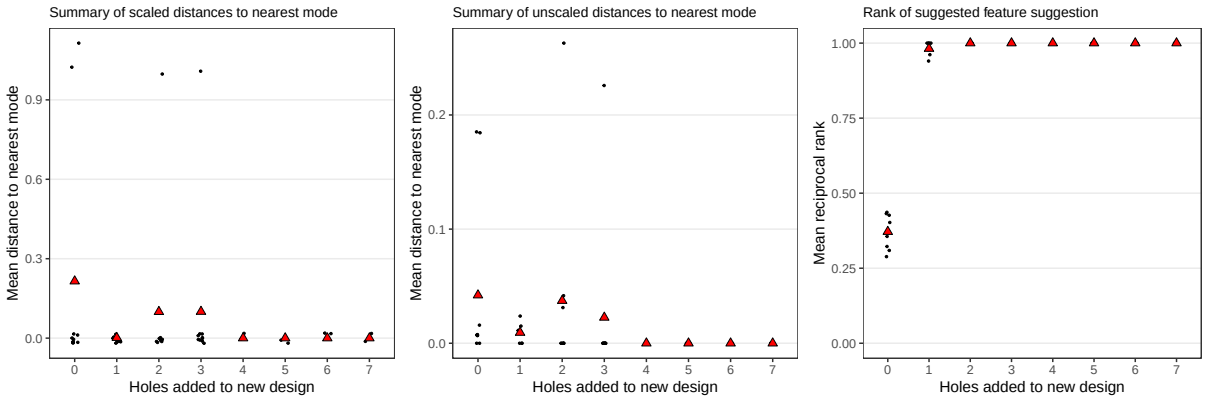



Figure 20: Cross-validation predictive performance. The x-axis indicates how many holes have been added to a new design and the y-axis records the average distance or rank. The black points give the within fold average measure and the red triangle the average performance across the folds.

# List of Figures