Interactive Model Prototyping in Visualization Space

O. Daae Lampe[1,2] and H. Hauser[2]

[1]Christian Michelsen Research, Norway
[2]Department of Informatics, University of Bergen, Norway

Abstract

Researching formal models that explain selected natural phenomena of interest is a central aspect of most scientific work. A tested and confirmed model can be the key to classification, knowledge crystallization, and prediction. With this paper we propose a new approach to rapidly draft, fit and quantify model prototypes in visualization space. We also show that these models can provide important insights and accurate metrics about the original data. Using our technique, which is similar to the statistical concept of de-trending, data that behaves according to the model is de-emphasized, leaving only outliers and potential model flaws for further inspection. Moreover, we provide several techniques to assist the user in the process of prototyping such models. We demonstrate the usability of this approach in the context of the analysis of streaming process data from the Norwegian oil and gas industry, and on weather data, investigating the distribution of temperatures over the course of a year.

Categories and Subject Descriptors (according to ACM CCS): I.3.6 [Computer Graphics]: —Interaction techniques G.3 [Probability and Statistics]: —Time series analysis

1. Introduction

Modeling is an essential part of scientific work. To be able to learn from observations and to utilize the gained knowledge for subsequent analysis, such as prediction, the modeling of the observed phenomenon in some sort of a prototype is crucial. Also, central to science is that model hypotheses are tested, refined and validated, or possibly rejected after testing. In the following, we consider a *model* to be a physical, mathematical, or logical representation of a system entity, a natural phenomenon or process, and that *modeling* is the act of creating a model [Sil01]. In experiments or, as we will focus on, modern process logging, measurements and data is gathered. Establishing a model on measured data often starts by employing empirical / statistical tests with trial and error, finally ending up with a *model prototype* and the statistical confidence on the model's accuracy. When and if the model prototypes hold up to scrutiny, one can aim to generalize these, and create a *model template*. The model template can be thought of as a scale invariant model, something that would fit to data irrespective to influencing factors, and then used to quantify these factors. E.g., a model template of time over height squared would, if applied to Galileo's raw experimental data, establish the gravity constant, along with the statistical confidence of this value.

Considering the visualization of particle paths in a tokamak (fusion reactor) as another example, we first consider that the most obvious footprint of direct data visualization is the fact that the particles intensely rotate – an observed phenomenon that is principally important, but not really surprising. To see this in a visualization might be interesting, but shortly after confirming the expected rotation of particles, we want to proceed and look behind this phenomenom: is there any secondary motion characteristic to be seen? To actually check such a hypothesis, we can aim at abstracting already understood and accepted aspects of the investigated phenomenon from the data visualization. This abstraction leads to three results: (a) the finding itself, which will undergo an externalization, where the finding is pulled out of the visualization represented in a different form, and (b) a residual data visualization – where the finding has been subtracted from – which then allows for studying remaining aspects of the observed phenomenon that do not follow the model. In the case of the tokamak example, we can think of an abstracted visualization of the particles, e.g., by using a Poincaré map (the main feature, i.e., the rotation of the particles, is then no longer visible, but only off-rotation deviations of the particle paths). This clears the view and allows the user to gain a more thorough understanding of complex phenomena through iterated analysis, including modeling and abstraction. (c) since large scale movements or densities of data are subtracted after the abstraction, this enables the further study of features which might be a magnitude smaller than the overshadowing and perhaps obvious features.

With this paper we aim to introduce a novel iterative workflow of assisted modelling, abstraction and subtraction to completely map a dataset from the originally visualized view to an abstracted and quantified one. To achieve this goal, we first provide a novel technique to assist a user to sketch locally optimal models, and then how to represent these in an abstract and quantified manner. Our technique is mainly applicable to data that can be mapped to a 2D representation.

The remainder of this paper is organized as follows: next we discuss some related work. Then we elaborate on the theoretical part of this work in Sec. 3, before we go into detail with respect to our technique in Sec. 4. In Sec. 5 we present results from the application of our approach and demonstrate its usefulness in this context.

2. Related Work

Extracting well defined features is a related topic often studied in the field of flow visualization. Post et al. [PVH*03] provides a good overview of the current state of art and how the features are found, abstracted, and quantified. On other phenomenons where the basic models is understood, data for visualization can often be reduced or abstracted. Löffelmann et al. [LKG98] use Poincaré maps as such a technique to create abstractions of data, reducing the dimensionality, and thus allowing the visualization of secondary features. On data in which the model is not understood, Rheingans and desJardins showed that inductive learning techniques, such as self organizing maps (SOM), can construct explanatory models for large, high-dimensional data sets [dR99, Rd00]. Their technique employs an overlay of models and data visualization, and thus creates an implicit visual comparison of model vs. data. Models as such are used in all sorts of scientific work – it is therefore perhaps beyond the scope of this work to reasonably discuss the role of models in science and visualization. Examples reach as far as into model-based segmentation of medical data (for example research for cardiac diagnosis [ZSBH08]) or into model-based object recognition (such as for robot vision [CD86], for example). These approaches show how models are used for classification and segmentation. Often, they also utilize a difference view, in which they show the match of a model to the data (which here relates to our residual data visualization). By iteratively adopting our technique, we can find higher order features, which are related to the field of multi-resolution analysis and multi-scale modeling, such as wavelet-based approaches [Wal04], for example. However, instead of focusing on a decomposition in frequency domain, we capitalize on partial abstraction which is feature-based and local.

The obvious step after extracting features is to put them to good use. Liu and Stasko [LS10] investigate how internal representations (mental models) and external visualizations relate to each other. The authors state that such mental models are used during visual reasoning to "simulate" the behavior of the corresponding visualization system [LS10]. Our approach helps the analyst to externalize such mental models, and compare the data to it. Shrinivasan and van Wijk and Yang et al. have investigated how to effectively support an *externalization* of findings in visualization. Yang et al. [YXRW07] describe a system which allows users to externalize findings, or *nuggets*, while exploring a dataset. These nuggets are then added to a *Nugget Management System*, where clustering and meta-information, help the sense making process. They also describe how visualization in this nugget space prove useful as an abstraction of the original data. Shrinivasan and van Wijk present the *Knowledge View* [SvW08] in which not only the findings are externalized, but also the interaction path that lead to it. Their user study shows favorable results with respect to externalizing knowledge in mind maps. Shrinivasan and van Wijk and Yang et al. have investigated how to effectively support an *externalization* of findings in visualization. Yang et al. [YXRW07] describe a system which allows users to externalize findings, or *nuggets*, while exploring a dataset. These nuggets are then added to a *Nugget Management System*, where clustering and meta-information, help the sense making process. They also describe how visualization in this nugget space prove useful as an abstraction of the original data. Shrinivasan and van Wijk present the *Knowledge View* [SvW08] in which not only the findings are externalized, but also the interaction path that lead to it. Their user study shows favorable results with respect to externalizing knowledge in mind maps.

The visualization scheme utilized in this work, is highly dependent on a frequency based technique that also supports meaningful difference views. Daae Lampe and Hauser presented a technique [DH11b] that enables the continuous distribution of data, using kernel density estimates (KDE). This technique also extends to support the continuous distribution of data-samples that is temporally connected, in that it creates a line-kernel that connects these samples. Daae Lampe et al. also effectively utilized these 2D KDE techniques for difference views [DKH10] in an application that aimed to generalize how to create multiple views that highlight the differences in distributions between distinct categories.

3. The Basic Idea

One of the goals of this research is to effectively support practitioners and scientists to analyze process data that is streaming in or updated at considerable rates. We provide an approach that allows users to rapidly prototype models

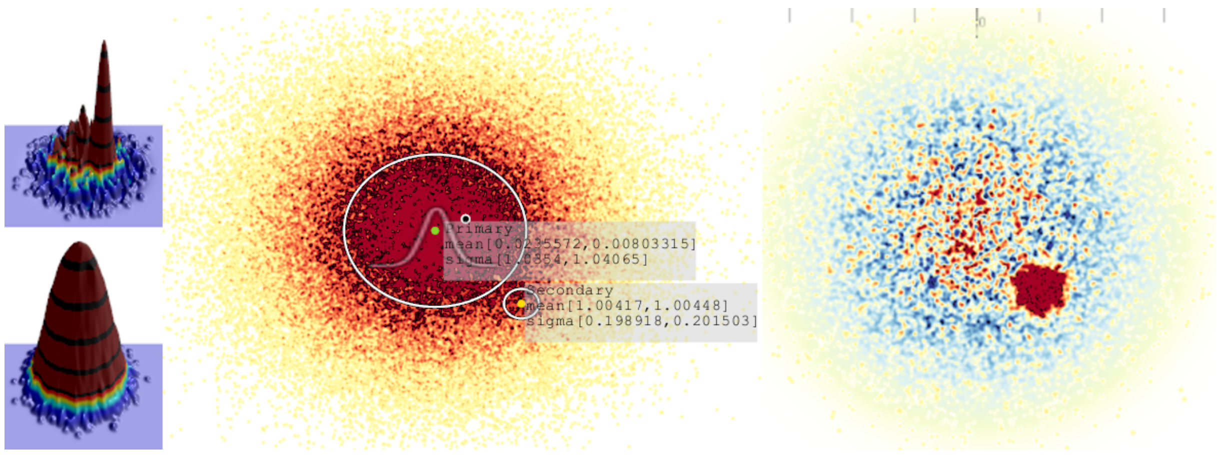

Figure 2: *A 2D version of the example in Fig. 3. The left images shows the (logarithmic) height-map after and before subtraction. The middle image shows the quantified measures as read out from both the primary and the secondary feature (after fitting two model prototypes). The image on the right shows the data, after abstracting the primary feature, clearly revealing the secondary feature, even though it was almost completely hidden.*

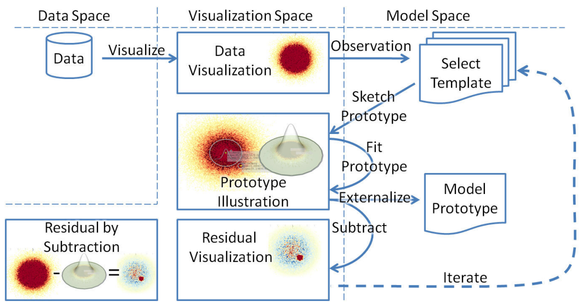

Figure 1: *Our proposed workflow: visualize and observe, sketch and fit, externalize and subtract, then iterate.*

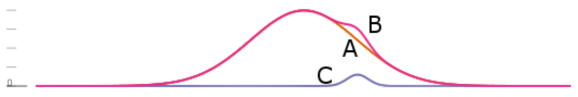

Figure 3: $A(x) = N(0,1)$, $C(x) = 0.05N(1,0.2)$ *and* $B(x) = A(x) + C(x)$

for structures which the user identifies in a visualization of the data. These model prototypes act (1) as parts of the externalization of the user findings and (2) as means to quantify the structures for subsequent user tasks. Accordingly, we first focus on how to identify structures which lend themselves to model prototyping. We see two opportunities: either the user has a conceptual model of what to look for (analytic/confirmative setting), or she/he aims at creating one by looking at the data (explorative setting). In the first case, it is useful to integrate the anticipated model within the visualization, to get initial information on how well the data fits the model. In the second case, it is useful to have a visualization that supports the user in interactively prototyping the model to then subtract it from the visualization, and get immediate feedback on how well it fits.

From this description we extract our workflow: *visualize and observe, sketch and fit, externalize and subtract, then iterate*, as shown in Fig. 1. This figure is read from top left then right or down. The data is visualized, and by observation an interesting feature is detected. The user selects a suitable model template and by sketching onto the visualization, creates an initial model prototype. Through further sketching and automatic fitting, the prototype is finalized. This complete prototype is externalized to model space, and a residual visualization is created by subtracting the model prototype from the data visualization. At this point the procedure can be repeated by observing another feature in the residual visualization, selecting another template to prototype, and so on. To explain by example, we consider a test dataset consisting of serveral random samples adhering to two different normal distributions. One of these distributions are of a magnitude smaller than the other one, and thus occluded. This dataset is shown in Fig. 2 and a simplified 1D version of the same is shown in Fig. 3.

Visualize and observe: we first draw all the data and the middle figure of Fig. 2 is shown. In this figure we observe only a single normal distribution, and thus decide that this would be a suitable candidate model.

Sketch and fit: in this second step, we pick the selected candidate model, the normal distribution, and mark its center near the observed center directly in the visualization. Immediatly when the model is placed, an automatic fitting will initiate and the sketched model will be optimized to fit the data.

Externalize and subtract: the parameters from the, now optimized, model are extracted and displayed. In our example case, the normal distribution model yielded a mean $\mu = (0.02, 0.01)$ and variance $\sigma = (1.035, 1.04)$, which is close to the reference $N([0,0],[1,1])$. These parameters can then be externalized, e.g., to a simplified table of extracted results. After this, the density of this externalized model is subtracted from the view, yielding the visualization to the right in Fig. 2.

Iterate: Finally, now presented with the view where the main feature is subtracted, we can start over again by observing the second feature as another normal distribution. After sketching and fitting this second normal distribution, we can extract the parameters $N([1.004, 1.004], [0.1989, 0.2015])$, here compared to the reference $N([1,1],[0.2,0.2])$.

4. Interactive Model Prototyping

In this section, we explore the details related to the proposed workflow, which are, visualization, model sketching and fitting, externalization or quantification, and interactive convergence.

4.1. Visualization

To support the proposed workflow we separate our visual aspects into three parts, the data visualization, the model prototype illustration, and the residual visualization. The model prototype illustration serves the purpose of giving a non-occluding and condensed view of where all the previous and the current model prototypes are located. Additionally, the prototype illustrations act as handles for interaction. The residual visualization serves several purposes. The first purpose is to show how well the model fits the data, and the second one is to then utilize the *visual range* better (through a scale-up operation). Human perception, and thus visualization, has a limited tolerance for range, e.g., there is finite limit to how many colors we can distinguish, or a limited range in how we can perceive brightness. By subtracting low frequency, high amplitude, features, we can effectively and automatically create a new and optimized range.

Streaming process data requires a direct in situ visualization of the data, since it is constantly updated. The visualization technique utilized here, is based on work by Daae Lampe and Hauser. [DH11b], and 2D Kernel Density Estimates (KDE). This technique visualizes a large set of samples, but displaying the convolved sum of kernels, one per sample, resulting in an analytical density function, that supports meaningfull difference views [DKH10]. Additionally, the usage of scaled kernels will create a visualization that shows the distribution of time, independent of sampling rate.

4.2. Model Sketching and Fitting

In computer science terms, a model template would be a class, and a model prototype would be an instance of such a class. In the process of creating a prototype, *sketching* is considered the manual input, and *fitting* the automatic algorithm assisting the user. Model templates come with properties, that the prototype needs to instantiate, which we categorize below. We consider them to be *shape*, *distribution*, and *scale*.

Shape characterizes the form of the model *along* the sequence of samples (after visualization). A linear structure can be described by a line model, more complex forms would follow spline curves, for example. In our case, we are fine with a piecewise linear model template. However, more complex models are equally possible (as long as a fitting procedure, see below, is available, as well). We refer to this central characteristic of a model as the *shape construct*. Selecting a shape requires the selection of the following parameters, *shape construct* and *control points*. We will only consider the following subset of shapes for the remainder of this paper: the single point construct (see Fig. 2), and the piecewise linear construct, wich will fit data with some correlation.

Distribution determines the form of the model *across* the sequence of samples. Whether it is due to noise, weak measurements, or other natural phenomena, real world data rarely ever line up perfectly. We therefore consider a certain data distribution across the sequence of data samples, which we model accordingly. The definition of the distribution we will refer to as the *distribution construct*. Selecting a distribution requires the selection of the following parameters, *distribution construct* and *width*. In the following we will denote this width as **r**, a vector separating the "radius" in the screen space coordinates u and v.

Scale, finally, is a measure of intensity. Depending on the visualization parameters, this parameter will have different meanings. E.g., for a box, the scale will be the average within it, for other, it will give a more complex measure of the intensity within the model.

Summing this up, we need to find a matching shape construct, a matching position, select a distribution construct, find the distribution width, and finally determine the scale, when we aim at fitting the model prototype to the data. To measure how well a model prototype fits the data, a problem not very different from image comparison, a correlation function like sum of squared differences has proven to be useful [GS99]. Other difference norms, such as the L1 norm, for example, also are possible and the choice of which norm to use is usually application-dependent. After choosing a squared differences norm (here L2), we investigate the opportunities to minimize it for fitting. To simplify the function to minimize we will consider the selection of shape construct and of distribution construct as selected manually by the user (according to his or her a priori assumptions about the data). In the following, we denote the discrete scalar field, which results from mapping the data to visualization space a $D(u,v)$, where u and v are the screen or canvas coordinates, and the models scalar field (also after mapping into visualization space) as $M(u,v)$. To generate this scalar field M, we first need to select the shape position $\mathbf{p}$, the distributions radius/extension $\mathbf{r}$, and the scale/height h. Based on these selections we have a function $L(\mathbf{p},\mathbf{r},h)$ which when mapped to the visualization space represents M

$$L(\mathbf{p},\mathbf{r},h) \rightarrow M(u,v) \tag{1}$$

Defining $\mathbb{UV}$ as the natural numbers from zero to canvas-width and canvas-height we get the difference measure (L2):

$$s = \sum_{(u,v)\in\mathbb{UV}} (D(u,v) - M(u,v))^2 \tag{2}$$

From eq. (1) and (2) we find that s, the sum of squared differences, is a function $s(\mathbf{p},\mathbf{r},h)$. From this we can extract our target variables through the following optimization:

$$\underset{\mathbf{p}\in\mathbb{UV},\ \mathbf{r}\in\mathbb{R}>0,\ h\in\mathbb{R}>0}{\operatorname{argmin}} \quad s(\mathbf{p},\mathbf{r},h) \tag{3}$$

Optimizing this equation is not straightforward, but since the user inherently sketches close to the desired solution, we can avoid potential problems with locating the global optimum, and only aim for the local minima.

We experienced satisfying results with the traditional Newton's method for this optimization problem, due to its good convergence [NW99] in local problems. Using Newton's method requires a Hessian matrix, a function that is twice differentiable, and an initial estimate x_0 that is sufficiently close to the solution. Calculating the Hessian matrix (or inverse thereof) is an computationally expensive operation, and we separate the derivatives and to minimize the function s by individually minimizing the variables in order of their influence of the overall model field

$$\underset{\mathbf{p}\in\mathbb{UV}}{\operatorname{argmin}}\, s(\mathbf{p}), \quad \underset{\mathbf{r}\in\mathbb{R}>0}{\operatorname{argmin}}\, s(\mathbf{r}), \quad \underset{h\in\mathbb{R}>0}{\operatorname{argmin}}\, s(h) \tag{4}$$

We look into these optimizations in Sec. 4.4 with measured results of convergence.

4.3. Quantification and Model Prototyping

After completing a number of model prototypes in visualization space, we transfer quantitative information back from visualization space into *model space*, a technique called externalization. Model space can be thought of as a summary of the understood features found in the data, and thus a more holistic approach to modeling is possible; one that also takes model parameters into account, which haven't dealt with in visualization space. For example, when visualizing speed vs. height of an object in free fall, this only can lead to model prototypes correlating those two parameters, not (yet) considering other potentially influencing factors, such as aerodynamic drag, etc. As established in Sec. 4.2, the information available is the position $\mathbf{p}$, distribution extent/radius $\mathbf{r}$, and scale h. Transferring $\mathbf{p}$ to model-space is trivial, and so is also $\mathbf{r}$. If the selected distribution construct is *box* or *linear*, then the transformed $\mathbf{r}$ is directly the radius around our shape construct. When using other distribution constructs we must allow for other interpretations of $\mathbf{r}$, e.g., the normal distribution, where variance or σ is more informative.

We have now described how to extract positions of our abstraction, its distribution width and a scale, based on a given intensity. We will look into more detail on how to apply this information in synthetic and real life cases in the next two sections, but we can already now see the usefulness in cases as pure statistical measures, or quantitative readouts of mean and variance.

4.4. Interactive Convergence

When sketching model prototypes, it is inherently hard to accurately or optimally draw the model prototype, as this would require the user to locate a local minimum based on several parameters. As introduced in Sec. 4.2, this would require the user to set five parameters, $\mathbf{p}$, $\mathbf{r}$, and h, when she/he is modifying a single point model construct. In this work we suggest an assisted prototype fitting, in which the user gets feedback on whether the first suggestion will converge towards his/her desired solution, or not. In the interactive mode, the user moves one point, and when it is released at its new position, the fitting algorithm will initiate. The iterative fitting algorithm is configured such that it will slowly converge/diverge at its first steps. If the initial suggestion was not sufficiently close to the optimal position $\mathbf{x}*$, it will diverge, away from the users desired target position. Instead, we iterate the fitting only a few steps, with constant step size instead of using Newton's method, so that the user can click and redirect the point before it *runs* away. When the user sees that the point is converging towards the desired solution, she/he can initiate the fitting algorithm that will then converge with the speeds that Newton's method offers.

As discussed in Sec. 4.2, we established that we need to compare the least squares of the model vs. the data. To initiate the fitting, we first need a rasterized version of the data. This texture is created once per frame, or when the dataset is updated. Next, we need a rasterized version of the model, which we create using a *construct aware* rasterize function, specific for the different implemented models. To recall, we considered the data's rasterized texture as $D(u,v)$ and the models, $M(u,v)$. Next, we need to calculate the least squares, and then we need to sum all the calculations for u and v. These calculations, as specified by Eq. (2), are implemented on a shader, which first performs the least squares, and secondly performs the reduction sum. We discussed the separation of the optimizing previously (see Eq.(4)), and as our tests have shown good convergence, when we iterate stepwise in our previously implied order (ie. one step with position, one with extent, and then with height, before reiterating

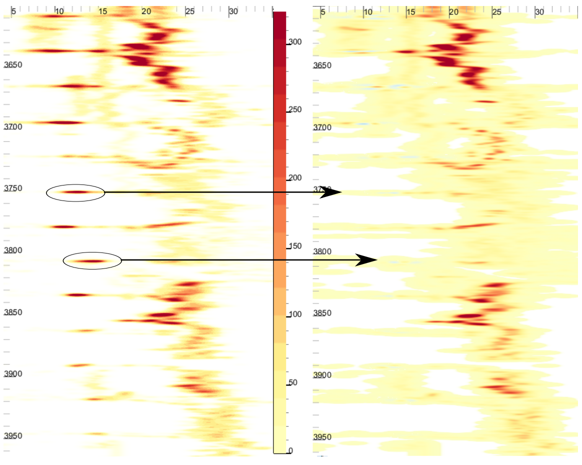

Figure 4: *Torque in kN.m over depth. The figure on the left shows the original data, containing some ROB tests we have identified, modeled and subtracted from the residual view to the right.*

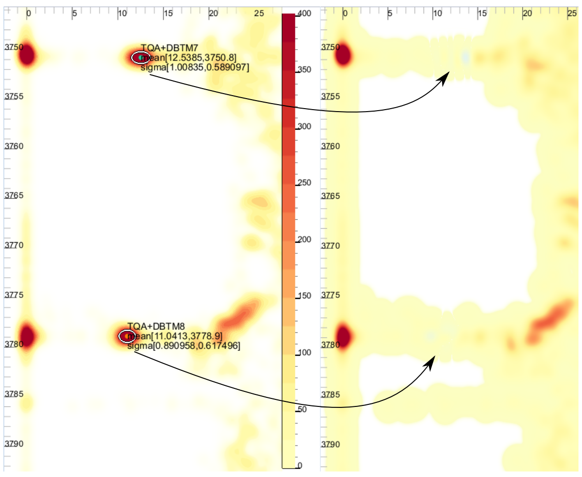

Figure 5: *A zoomed in view from Fig. 4 onto two rotation off bottom tests, showing how well this data is modeled and removed from the residual view. The residual view to the right has a larger area of yellow values due to a dynamic range color re-scale.*

next step). If we have a point construct, we repeat the process of calculating the sum of least squares, for our position $\mathbf{p}$ in the positions:

$$\mathbf{p}+\Delta u, \quad \mathbf{p}-\Delta u, \quad \mathbf{p}+\Delta v, \quad \mathbf{p}-\Delta v$$

Next, for Newton's method we need $f'(\mathbf{p}_n)$ and $f''(\mathbf{p}_n)$, now let (for both u and v):

$$f'(\mathbf{p}_n) = \big(s_n(\mathbf{p}_n+\Delta) - s_n(\mathbf{p}_n-\Delta)\big)/2|\Delta| \qquad (5)$$

$$f''(\mathbf{p}_n) = \big((s_n(\mathbf{p}_n+\Delta) - s_n(\mathbf{p}_n)) - (s_n(\mathbf{p}_n) - s_n(\mathbf{p}_n-\Delta))\big)/|\Delta| \qquad (6)$$

And thus we are able, to calculate $\mathbf{p}_{n+1} = \mathbf{p}_n - \frac{f'(\mathbf{p}_n)}{f''(\mathbf{p}_n)}$, $n \geq 0$.

Next, we fit the extension of our distribution $\mathbf{r}_n$ (in the case of normal distribution, this would be σ), ending up with $\mathbf{r}_{n+1}$, finally, before calculating the scale, or height h_n of the distribution. This we implement in a similar manner, by calculating $s_n(h_n)$, $s_n(h_n+\Delta)$, and $s_n(h_n-\Delta)$, and creating the derivatives, using the same procedure as above, then by Newton's method, we calculate a new height h_{n+1}. To extend the above method, that was defined for point constructs, for piecewise lines, we repeat the first step, finding $\mathbf{p}_{n+1}$ for all $\mathbf{p_n}$ in the piecewise line segment.

5. Case Studies

We now present two case studies, one on process data from the Oil and Gas sector on real-time data generated under drilling operations, and one analyzing the temperature changes through several years.

5.1. Process Data

We will apply our approach to a dataset that contains 116,191 time-steps in total, spanning over a period of approx. 28 hours, with a varying sample rate from $1/30Hz$ to $30Hz$. It is a multivariate dataset containing 25 variables at each time-step in three major categories, *measured* data from the surface, measurement while drilling (*MWD*) equipment and *derived* data. MWD or down-hole measurements are measured from MWD tools down in the well and then transmitted to the surface via mud pulse.

A prominent usage of these data-streams are logging, early detection and warning in case of incidents or analysis to elaborate on a problem evolving or past. To give an early warning on potential incidents, a common strategy applied are friction tests. Increased friction can be a good indicator on amassed cuttings in the hole. To infer friction, two different techniques are applied: torque based tests, and weight based tests. When pulling the string up we can expect friction to act as a force against the movement, thus increasing the measured weight, and similarly when moving the string down we expect a lower weight. Rotating the drill string will give a response torque measured on the surface. This torque will increase with increasing friction. This test is called *rotation off bottom* or ROB.

Fig.4 shows abstracted and residual data visualization representing ROB. Notice how all the higher densities (representing time spent performing this ROB), are removed in the residual view to the right. Fig.5 shows a zoomed in version of Fig.4, where two of these tests and additionally their quantitative parameters are shown. From this model proto-

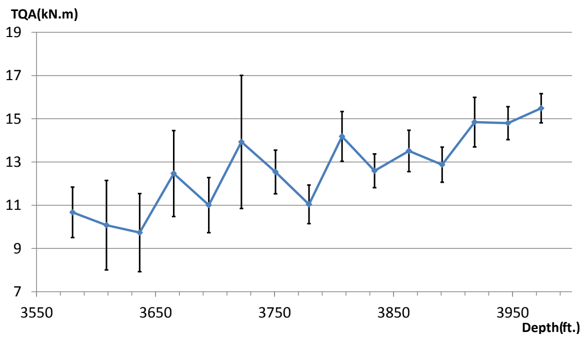

Figure 6: *Changing torque in kN.m over depth in feet, for a series of ROB tests. The abstracted results from Fig.4 are shown in a graph with error bars at 1σ for both depth and torque uncertainty. In this figure, the uncertainty for depth is negligable.*

type, one can now read out the average and mean during this ROB. A big advantage our technique has compared to the existing one, where the mean of all measured values during the ROB is taken, is that our technique would show a poor fit, if a poor ROB is performed. An example of a poor ROB is if the samples are increasing, or decreasing during the entire ROB period. Another poor ROB would have its samples clustered at two distinct torques, and a mean would then be misleadingly in middle between them. After matching a total of 15 of these ROB tests in Fig.4 we go to a more abstract view, where all the data is removed, and only the modeled statistcs are left. Fig.6 shows 15 ROB tests, with their standard deviations shown as error bars for both depth and torque. This abstraction illustrates quite well a problematic ROB friction that occurred at 3722 and at 3815 feet, but was put under control at larger depths with more moderate torque and a lower deviation (more certain measurements).

Quantifying torque is an important step in knowing how the down-hole conditions are developing, but that is only one step on the way of getting indications on what the friction is. There are no accurate algorithms that exist today, that can accurately calculate friction, even if all different conditions are taken care of. This is the reason why we look at torque, since it is an indication of friction even though many more variables affect the result.

5.2. Temperature

In this section we inspect hourly temperature readings from a single weather station, over the course of ten years, courtesy of eKlima [Nor]. The top of Fig. 7, displays the curve density estimate [DH11a] of these curves over an average year. In this display the most prominent feature is the seasonal change, with high temperatures during the summer, and lower temperatures during the winter. Fig. 7 also features an orange line overlaid to display the cyclic moving

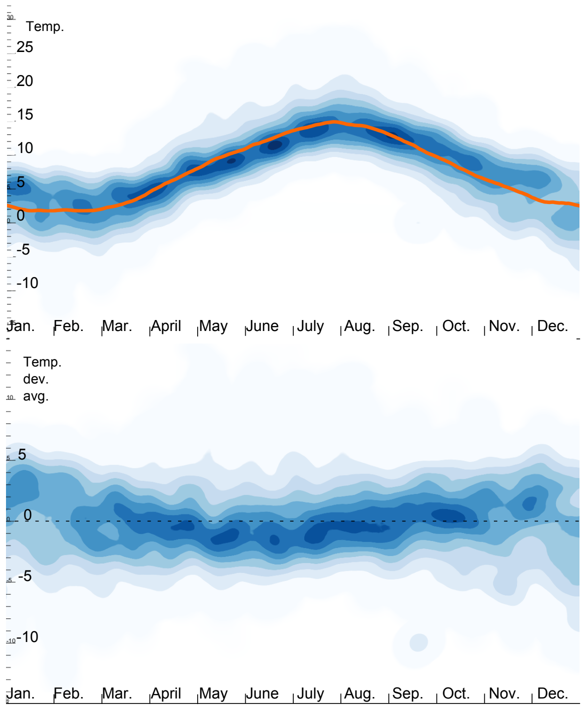

Figure 7: *On top, the distribution of measured temperature over the average year based on hourly data for ten years, by using the curve density estimate [DH11a]. The orange curve shows the cyclic moving average of the temperature over all ten years. On the bottom, the same data is shown, but here de-trended, by subtracting the moving average.*

average of the yearly temperature for these ten years. Such a moving average is a good representation and abstraction for the yearly temperature, given that the data follows a normal distribution, but might provide false information if not [DH11a]. To investigate how well this average can abstract the data, we first create a new dataset containing the differences between the moving average and the measured temperature. This new dataset of deviations from the moving average, is shown as the bottom graph in Fig. 7. The higher peaks, at approximately zero, during the summer months in this dataset, indicate a more stable temperature, i.e., the average temperature is measured more often. To investigate how well the normal distribution fits the de-trended data, we apply a linear normal-distributed model to it. The resulting difference view between the applied model and the de-trended data is shown in figure 8. Now, as opposed to the previous two figures, the focus is placed on the deviations from the normal distribution. Our attention is drawn to the high intensity above the norm in January and December. Since these intensities are red, they present areas where the measured value is higher represented than the normal distribution. However, since the average is placed at zero, it indicates

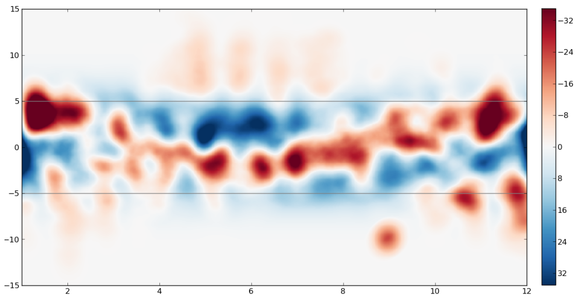

Figure 8: *The difference between a linear model and the detrended data from Fig. 7. The linear model applied has its mean μ on $y = 0$, a $\sigma = 1.85$ and its upper and lower quartile shown as grey lines. Note that due to the diffence view, the deviations shown here are of one magnitude greater than in Fig. 7.*

a "tail" of low temperatures dragging the average down, i.e., a negatively skewed distribution. A second finding here is the anomaly placed mid September, where we find a peak of overrepresented values at an extreme ten degrees below the average. After closer inspection in the dataset, we found that this represents missing values which was defaulted to zero. As a third finding, we point to the light red areas above the grey quartile line in the summer months. These indicate that during these months the actual distribution has a positive skewness, leading to a bigger "tail" towards higher temperatures than the average and standard deviation would account for.

6. Discussion, Conclusions, and Future Work

Looking at the results from our synthetic test first, we see that not only primary features are properly detected on rasterized data, but also secondary features. From Fig. 2 we measure that one unit in data space corresponds to 85 pixels, which means that if we can detect, with this exact resolution, close to $1/85 \approx 0.012$ units accuracy, then we have sub pixel accuracy.

Looking at the results in table 1 we refer to the results on the secondary feature. These results estimate the original model with not only sub-pixel precision, but with at a tenth of a pixel accuracy, on the variance. Further we see that the secondary feature's mean is estimated to a level of a third of a pixel, also sub pixel precision. On the primary feature, we see that the results are within pixels with regards to the reference, with the estimates a little on the high side. The primary feature is the first one fitted in our data visualization, and thus it includes the secondary feature in its estimate, something that can explain the pixel offset in μ. The results on accuracy are very promising, which is very interesting considering that our approach achieves these results at $O(n)$ (in time complexity, as we rasterize n points once).

Name	Value	Reference	Pixel err.
Primary μ X	0.02	0	1.7
Primary μ Y	0.01	0	0.85
Primary σ X	1.035	1	2.98
Primary σ Y	1.04	1	3.4
Secondary μ X	1.004	1	0.34
Secondary μ Y	1.004	1	0.34
Secondary σ X	0.1989	0.2	0.0936
Secondary σ Y	0.2015	0.2	0.128

Table 1: *Error measurements in data space units and pixels on standard normal distribution with secondary feature, see Fig. 3 and Fig. 2. A rendering with 85 pixels per unit has been used for these calculations.*

An important characteristic of our approach is the high degree of interactivity. When displaying streaming data, it is important to have a visualization scheme that is able to handle large time windows, i.e., if data is streamed one needs at some point to either omit “old” data from the visualization, or support a multi resolution scheme. We have implemented visualization mappings that allow fast rendering ($>$ 60 fps), even if we show datasets spanning several days ($>$ 200k samples). The feedback on convergence (or divergence) is also an important aspect that facilitates interactive analysis.

With this work we have demonstrated how data visualization can benefit from interactive model prototyping, externalization and subtraction so that expert users can rapidly proceed through an in depth analysis of streaming process data, following the *visualize and observe, sketch and fit, externalize and subtract, then iterate* pattern. Subtracting identified features from the data visualization allows the user to reveal secondary features and additionally results in an externalized prototype giving quantification and overview.

We have shown that interactive model prototyping in visualization space can accurately quantify measured data. Moreover, we have shown that an analyst can quickly compare suggestions for formal models, by bringing them into the visualization, perform prototyping, and get quantitative results on how well they fit. Another important part of our work has been to move visualizations beyond the initial discovery, and to give the users a view into secondary features. A general conclusion from our work is that application processes usually do not stop after discoveries in visualization and that is therefore important for visualization research to more intensely think about what has to follow visualization, e.g., externalization, quantification and ultimately action.

In future work, we plan to look further into different reconstruction techniques, and also different distributions. An interesting aspect would be to investigate the support for distributions with rotations or shear, by enabling support for a full covariance matrix, instead of the vector **r**. Another plan is to extend the piecewise linear model to support higher-order templates, like spline curves. It would interesting to consider an extension to multivariate fields, or even to three dimensional fields, using 3D rasterizing functions.

7. Acknowledgements

The work presented here is a part of the project "e-Centre Laboratory for Automated Drilling Processes" (eLAD), participated by International Research Institute of Stavanger, Christian Michelsen Research and Institute for Energy Technology. The eLAD project is funded by grants from the Research Council of Norway (Petromaks Project 176018/S30, 2007-2011), Statoil ASA and ConocoPhillips Norway.

References

[CD86] CHIN R., DYER C.: Model-based recognition in robot vision. *ACM Comp. Surveys (CSUR) 18*, 1 (1986), 67–108. 2

[DH11a] DAAE LAMPE O., HAUSER H.: Curve density estimates. *Comp. Graphics Forum 30*, 3 (2011), 633–642. 7

[DH11b] DAAE LAMPE O., HAUSER H.: Interactive visualization of streaming data with kernel density estimation. In *Proceedings of the IEEE Pacific Visualization Symposium (PacificVis 2011)* (March 2011), pp. 171–178. 2, 4

[DKH10] DAAE LAMPE O., KEHRER J., HAUSER H.: Visual analysis of multivariate movement data using interactive difference views. In *Proceedings of Vision, Modeling, and Visualization (VMV 2010)* (2010), pp. 315–322. 2, 4

[dR99] DESJARDINS M., RHEINGANS P.: Visualization of high-dimensional model characteristics. In *Workshop on New Paradigms in Information Visualization and Manipulation* (1999), pp. 6–8. 2

[GS99] GESÙ V. D., STAROVOITOV V. V.: Distance-based functions for image comparison. *Pattern Recognition Letters 20*, 2 (1999), 207–214. 4

[LKG98] LÖFFELMANN H., KUČERA T., GRÖLLER E.: Visualizing Poincaré maps together with the underlying flow. In *Mathematical Visualization - Algorithms, Applications and numerics*. Springer, 1998, pp. 315–328. 2

[LS10] LIU Z., STASKO J.: Mental Models, Visual Reasoning and Interaction in Information Visualization: A Top-down Perspective. *IEEE Trans. Visualization and Computer Graphics 16*, 6 (2010), 999–1008. 2

[Nor] NORWEGIAN METEOROLOGICAL INSTITUTE: eKlima. `eklima.met.no`. [Online; accessed Nov-2010]. 7

[NW99] NOCEDAL J., WRIGHT S.: *Numerical Optimization*. Springer, 1999. 5

[PVH*03] POST F. H., VROLIJK B., HAUSER H., LARAMEE R. S., DOLEISCH H.: The State of the Art in Flow Visualization: Feature Extraction and Tracking. *Computer Graphics Forum 22* (2003), 775–792. 2

[Rd00] RHEINGANS P., DESJARDINS M.: Visualizing high-dimensional predicitive model quality. In *IEEE Visualization* (2000), pp. 493–496. 2

[Sil01] SILVERT W.: Modelling as a discipline. *International Journal of General Systems 30*, 3 (2001), 261–282. 1

[SvW08] SHRINIVASAN Y. B., VAN WIJK J. J.: Supporting the analytical reasoning process in information visualization. In *CHI '08: Proc. of SIGCHI on Human factors in computing systems* (2008), pp. 1237–1246. 2

[Wal04] WALNUT D. F.: *An Introduction to Wavelet Analysis*. Springer, 2004. 2

[YXRW07] YANG D., XIE Z., RUNDENSTEINER E. A., WARD M. O.: Managing discoveries in the visual analytics proc. *SIGKDD Explor. Newsl. 9*, 2 (2007). 2

[ZSBH08] ZAMBAL S., SCHÖLLHUBER A., BÜHLER K., HLADUVKA J.: Fast and robust localization of the heart in cardiac MRI series. *Proc. of Int. Conf. on Computer Vision Theory and Applications* (2008). 2