CEE 4770 Final Project

Simulating Wind Loads on High-Rise Buildings

Submitted to: Prof. Mircea Grigoriu
December 21, 2019

Alex Dzieman (awd55@cornell.edu)
Ananya Gangadhar (ag757@cornell.edu)

Table of Contents

1. Introduction

1.1. Problem

Due to climate change caused by global warming, many parts of the world are experiencing more frequent extreme weather events like floods and hurricanes. Since the number of extreme weather events is predicted to only increase in the future, and with rapid urbanization leading to the construction of more and more tall buildings, it is of utmost importance to examine public infrastructure and buildings and predict their performance in such extreme events. This study models the wind loads on high-rise buildings using wind pressure coefficient data from wind tunnel experiments. In such wind tunnel experiments, a scaled down model of the building of interest is fitted with pressure taps on all faces and is subjected to varying wind speeds. Pressure coefficients for each pressure tap are recorded as a function of time.

The pressure coefficient is a dimensionless number describing relative pressures through a fluid flow field, and is described using the following equation,

$$C_P = \frac{p - p_\infty}{\frac{1}{2}\rho_\infty V_\infty^2} = \frac{p - p_\infty}{p_0 - p_\infty}$$

where p is the static pressure at a given point, p_∞ = free stream static pressure, p_0 = stagnation pressure in the free stream, ϱ_∞ = fluid density, V_∞ = freestream velocity of fluid.

By creating a model to simulate wind loads or pressure coefficients for various exceedance levels, we can better understand the effect of dynamic loads on high rise buildings, and predict where and when structural failures may occur.

1.2. Literature Review

Current wind load codes were written in the previous century, before the advent of advanced computational practices. This has lead to outdated wind load estimation that can have differences between measured values and estimated maximum values of over 40% (Simiu, Emil, and DongHun). Additional work done by the Department of Statistics of the University of Sao Paulo in Brazil has shown that wind speeds can be best modeled using the GEV distribution (Pinheiro,Elaine C., and Silvia L. P. Ferrari).

2. Data

2.1. Sources

Time series data from pressure taps in wind tunnel experiments were obtained from the Tokyo Polytechnic University Aerodynamics of High-Rise Buildings database (Wind Effect). The model scale is 1/400. The chosen high rise building has dimensions H:B:D = 5:3:1 and the depth of the experimental building model is 0.1 m. The angle of wind exposure to the front face of the building was set to 0°, and the wind exposure factor α = ¼.

There were 480 pressure taps distributed evenly along the front, sides and back of the high-rise building model and data was collected over 32.768 seconds with a frequency of 1000 observations per second. Time series data for all pressure taps was downloaded from the website. The source of time series data, wind velocity profile and pressure tap distribution are included in the appendix.

2.2. Relevant statistics

2.2.1. Moments

By analyzing the four moments (mean, standard deviation, skewness, kurtosis), we can estimate the distribution of pressure coefficients for each tap. Using MATLAB, the estimated mean, standard deviation, skewness and kurtosis of data for each pressure tap was calculated and plotted on contour maps of the building.

The **mean** or expected value of a distribution, denoted by E[X], is the central value of a dataset with n-values and is estimated as follows,

$$\text{Mean estimate} = \bar{x} = \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right) = \frac{x_1 + x_2 + \cdots + x_n}{n}$$

The **standard deviation**, denoted by σ, is a measure of dispersion of the values of a dataset of n-values from its mean and is defined as follows,

$$\text{Standard deviation} = \left(E\left[X^2\right] - E[X]^2\right)^{1/2}$$

$$\text{Estimated standard deviation} = s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2}$$

The **skewness** is the normalized third moment of a probability distribution and is a measure of its asymmetry about the mean. For a unimodal distribution, positive skew usually indicates a right tail of the distribution curve, while a negative skew indicates a left tail. Shown below is the mathematical definition of skewness, and its method of moments estimate for a dataset with n-values (used by MATLAB to estimate the skewness of vector data).

$$\begin{aligned} \text{Skewness } &= \frac{E[x-\bar{x}]^3}{\sigma^3} \\ \text{Skewness estimate } &= \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^3}{(\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^3})} \end{aligned}$$

The normalized fourth moment, or **kurtosis**, is a measure of how outlier-prone a probability distribution is (Pinheiro,Elaine C., and Silvia L. P. Ferrari). The kurtosis of a Guassian or normal distribution is 3. Distributions with more outliers than the normal distribution have kurtosis greater than 3, while those with fewer outliers have kurtosis less than 3. Shown below is the mathematical definition of kurtosis and the numerical estimation of kurtosis (used by MATLAB).

$$\begin{aligned} \text{Kurtosis} &= \frac{E(x-\mu)^4}{\sigma^4} \\ \text{Kurtosis estimate } &= \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^4}{\left(\frac{1}{n}\sum_{i=1}^{n}(x_i-x)^2\right)^2} \end{aligned}$$

The number of samples needed to estimate skewness and kurtosis accurately is much higher than that needed for mean or standard deviation due to the higher powers in the equation. Thus we used MATLAB to plot all the moment estimates of the first five taps as a function of sample size (shown in Figure 1) to ensure convergence and minimal noise interference for the given number of samples, 32768.

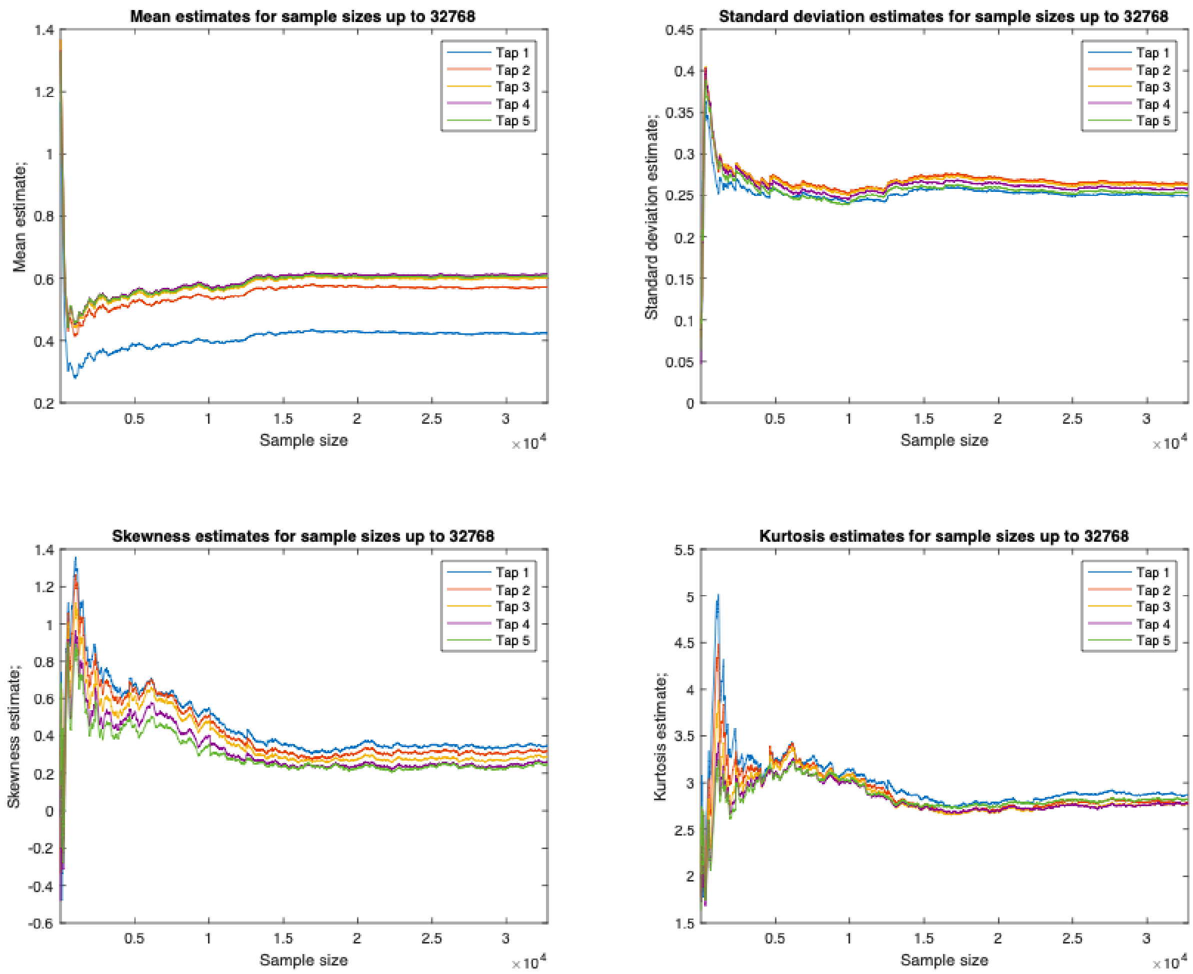


Figure 1. Graphs of (clockwise) mean, standard deviation, kurtosis, skewness of the first 5 pressure taps plotted against increasing sample sizes up to the actual value of 32768.

From the above figure, it is evident 32768 samples are sufficient to obtain a reasonably accurate estimate of skewness and kurtosis since the values start to converge rapidly after 15000 samples.

2.2.2. Correlation

The correlation is necessary for our analysis because it will allow us to see how much influence a pressure at any given point in time has on the pressure at a different point in time. The correlation between two random variables is defined below:

$$\mathrm{Corr}(X_1, X_2) = r_{X_1X_2} = \frac{E(X_1 - \mu_{X_1}) \cdot E(X_2 - \mu_{X_2})}{\sigma_{X_1} \cdot \sigma_{X_2}}$$

The correlation can be solved numerically for a pair of data sets (X_1, X_2) by the following equation:

$$r_{X_1X_2} = \frac{1}{n}\sum_i \frac{(x_1 - \mu_{X_1}) * (x_2 - \mu_{X_2})}{\sigma_{X_1} * \sigma_{X_2}}$$

With the above equation, correlation could be expressed as a function of the time between data points. Below is an image showing a pair of data points that would be paired to find the correlation for a given time "s" between data points. An important result from this analysis is that when the time separation is zero, the correlation is comparing the same data set and is 1.

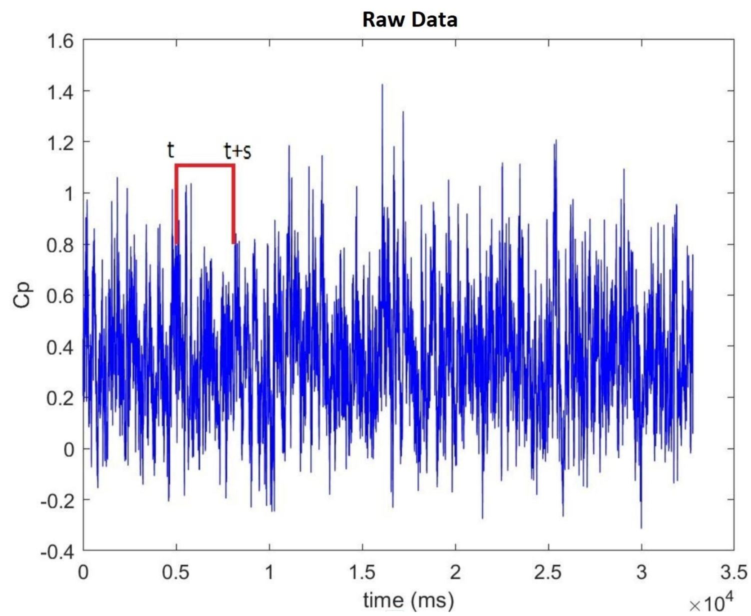


Figure 2. Time series data showing two points that would be compared for correlation.

A typical correlation function is shown below (Figure 3).

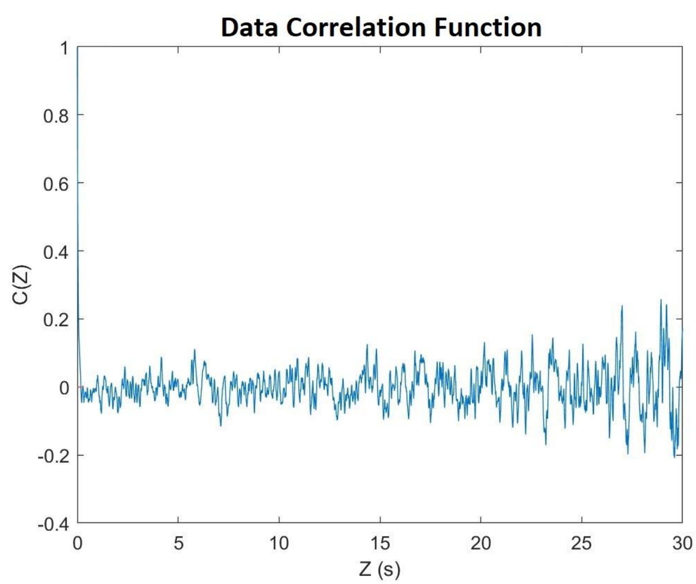


Figure 3. Typical correlation function for any given tab.

As the plot suggests, the correlation between points for time large differences can be assumed zero due to the large amounts of noise. Additionally, the false increase in correlation shown after 20 seconds can be explained by the fact that the amount of data to perform the correlation decreases the larger the time separation is, this is due to the data set being ~33 seconds long. To get true meaning from correlation function we must look at the first couple hundred milliseconds shown below:

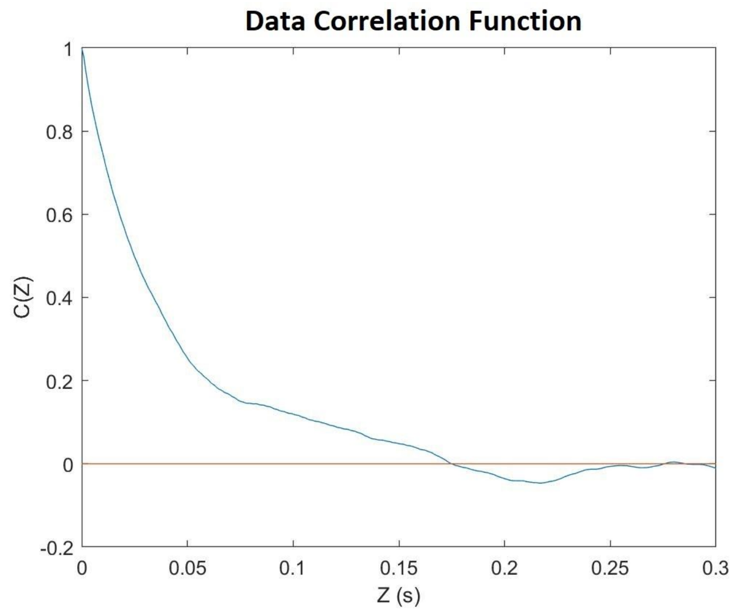


Figure 4. Typical Correlation function for any given tab.

The blue line is the correlation function for the data and the red line is a line on Y=0. As we can see the correlation starts at one, as expected, and steadily decreases to zero within the first ~180 milliseconds.

3. Methods of Analysis

3.1. Data fitting

In order to create a model for wind loads, estimating the distribution of the time series pressure coefficient data was necessary. A preliminary review of the dataset revealed each pressure tap histogram to have a unimodal distribution with varying degrees of asymmetry (Figure 5).

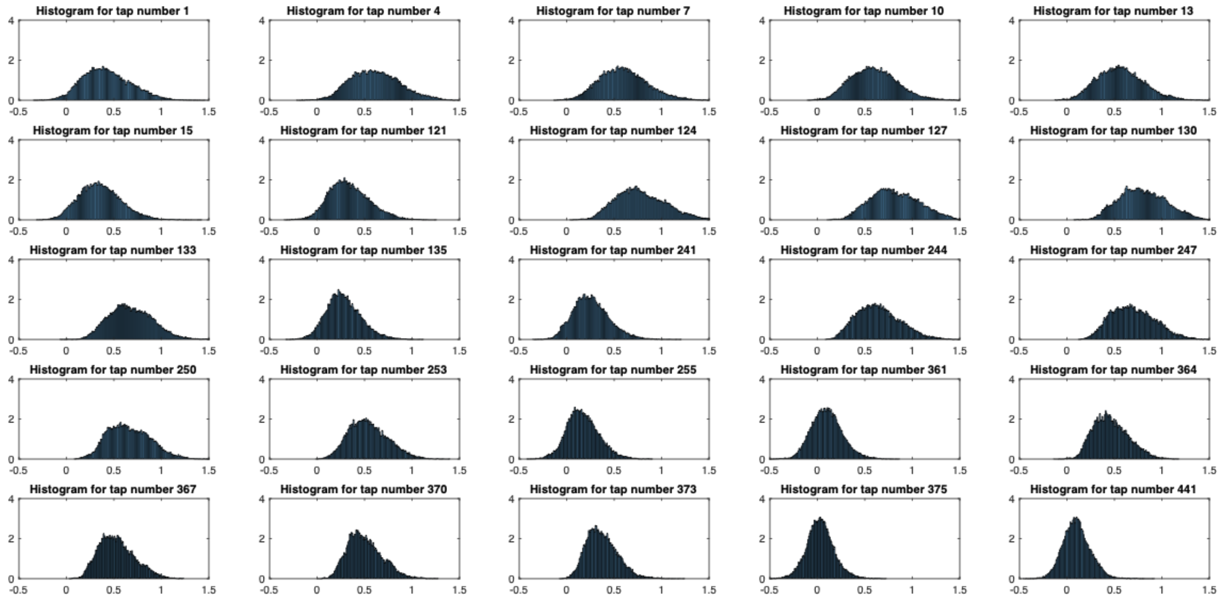


Figure 5. Histogram plots of various pressure taps on the front building face scaled to the same y-axis limits.

The following continuous probability distributions were tested to determine the best fit — normal, gamma, Gumbel, Generalized Extreme Value. Weibull and lognormal distributions were excluded since they require positive, real values but pressure coefficients can often take negative values.

3.1.1. Contour maps of moments

Using MATLAB, a contour map for each moment — mean, standard deviation, skewness and kurtosis — was generated to visualize the overall distribution of pressure coefficients for each pressure tap (Figure 6-9).

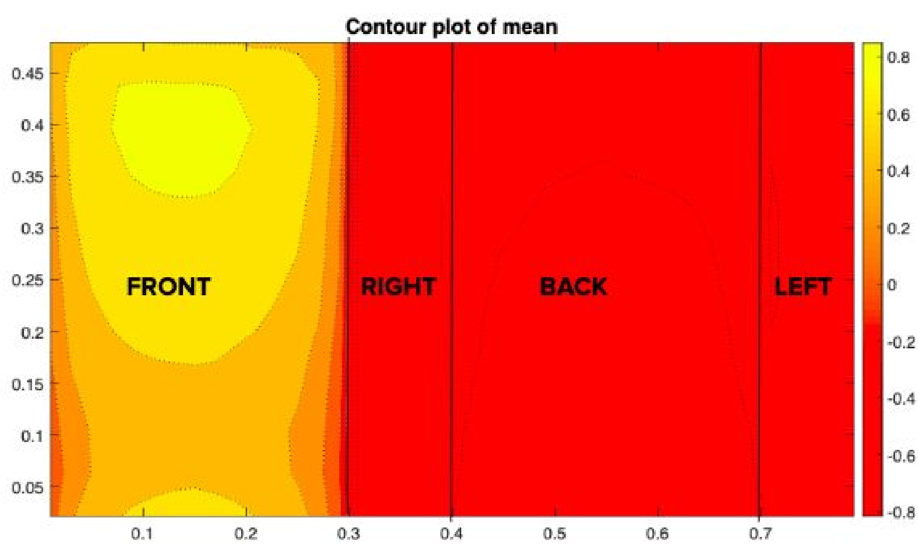


Figure 6. Contour plot of building faces showing mean of data from each pressure tap.

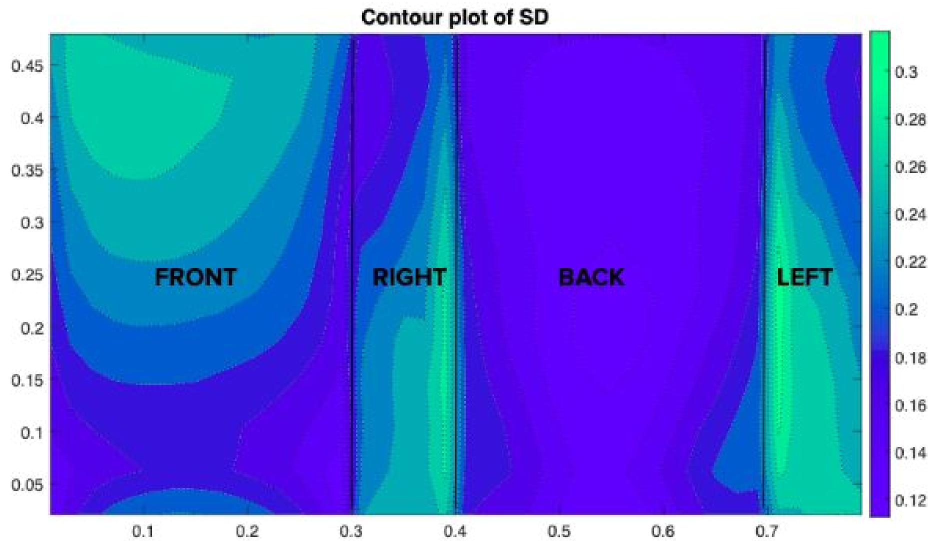


Figure 7. Contour plot of building faces showing standard deviation (SD) of data from each pressure tap.

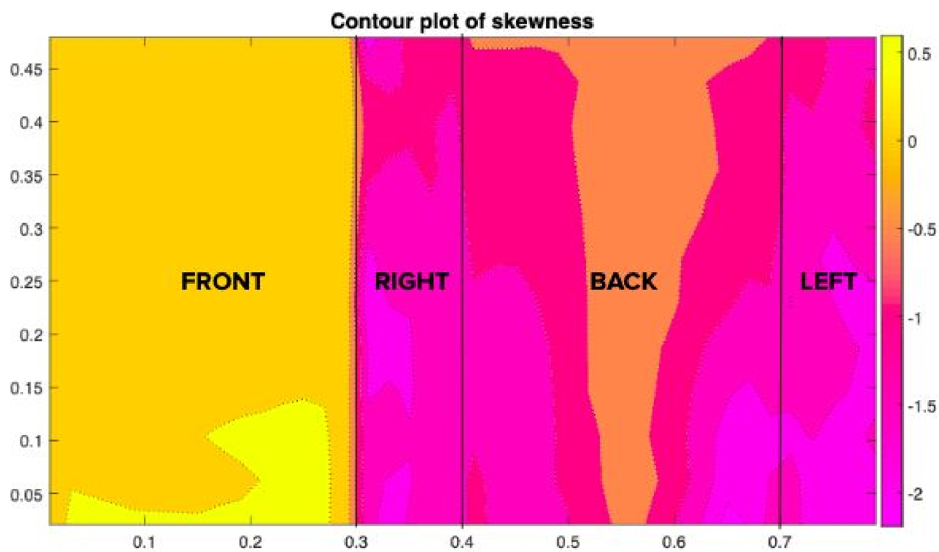


Figure 8. Contour plot of building faces showing skewness of data from each pressure tap.

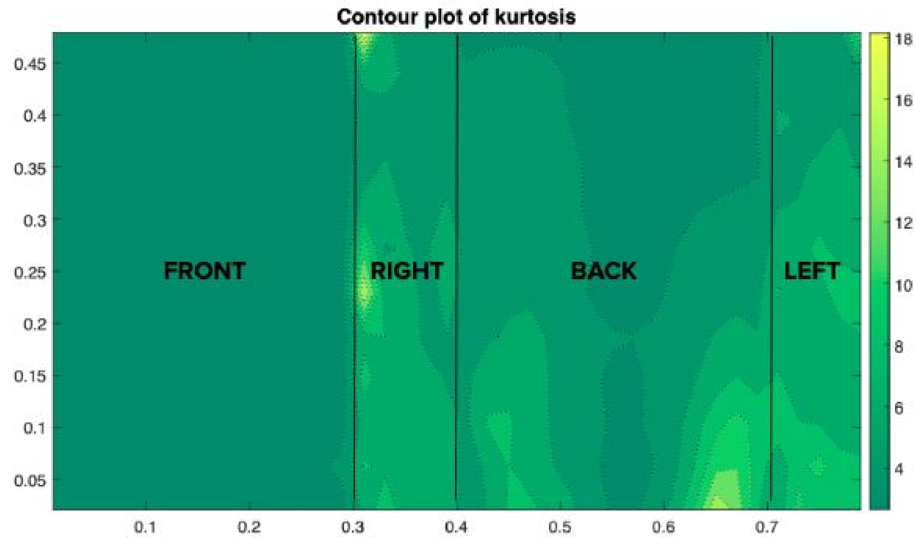


Figure 9. Contour plot of building faces showing kurtosis of data from each pressure tap.

While the front face has a small positive skew, the sides and back have larger negative skews (Figure 8). Interestingly, the kurtosis values vary from 2 to 18, especially on the sides and back (Figure 9). The Gumbel distribution has a fixed kurtosis of 5.4, and thus is not a good model for the vast majority of pressure taps on the sides and back of the high-rise building. The front face however has a kurtosis of around 3 which means the distribution at each tap is close to normal.

3.1.2. Generalized Extreme Value Distribution

Previous literature suggests that the Generalized Extreme Value (GEV) Distribution is the best choice for modeling wind loads (Pinheiro,Elaine C., and Silvia L. P. Ferrari). The GEV is a continuous extreme value probability distribution that combines Gumbel (Type I), Frechet (Type II) and Weibull (Type III) distributions. The GEV distribution has three parameters — location (u), scale (k), shape (α). The GEV distribution is Gumbel if α = 0, Weibull if α < 0, and Frechet if α > 0. Shown below is the mathematical notation used to define the GEV distribution in MATLAB.

$$y = f(x|\alpha, u, k) = \left(\tfrac{1}{k}\right) \exp\left(-\left(1+\alpha\tfrac{(x-u)}{k}\right)^{-\frac{1}{\alpha}}\right)\left(1+\alpha\tfrac{(x-u)}{k}\right)^{-1-\frac{1}{\alpha}}$$

$$\text{for } 1+\alpha\frac{(\mathrm{x}-u)}{k} > 0$$

The GEV distribution is more versatile than other extreme value distributions since it its skewness and kurtosis can vary as a function of its shape parameter α, and Monte Carlo simulations prove that it can fit wind load data well even within a 90% confidence interval (Pinheiro,Elaine C., and Silvia L. P. Ferrari). Additionally, there are several MATLAB functions that support operations with the GEV distribution. Thus we hypothesize that the GEV distribution will fit the data the best.

3.1.3. Q-Q plots

Since the data for a good number of pressure taps are close to normal, it is difficult to tell simply by looking at the various probability distributions which one is the best fit (Figure 10). A better graphical tool is thus needed to determine which distribution fits the data best. A Q-Q plot, or quantile-quantile plot, is a scatterplot created by plotting two sets of quantiles against each other. Should both sets have the same distribution, the points will form a straight line. Q-Q plots were made for pressure taps from all faces of the building by plotting the actual data against the theoretical distribution being tested. The straighter the line, the better fit the distribution is to the data.

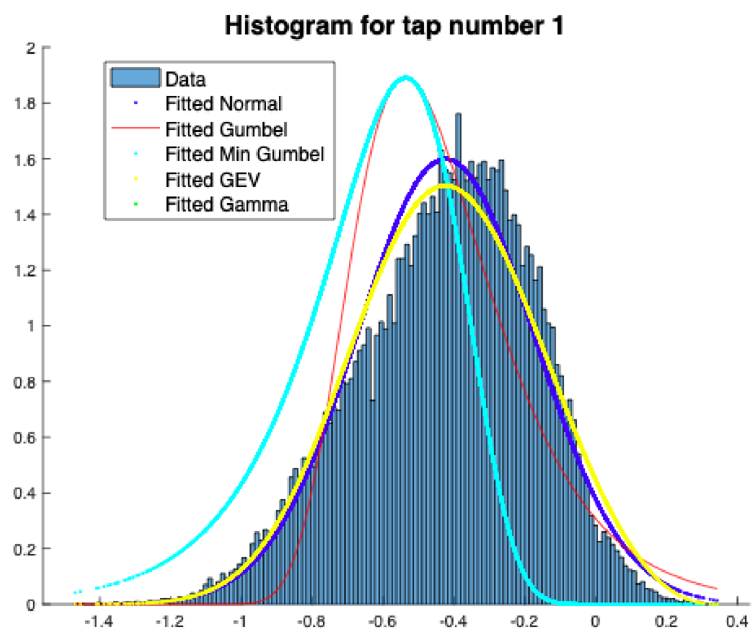


Figure 10. Example of a histogram for a pressure tap with overlapping distributions (normal, GEV), facilitating the need for Q-Q plots.

3.2. Generating Model and Samples

3.2.1. Analytical derivation

To model our data we made use of the standard normal random function whose equation is shown below.

$$G(t) = \sum_{k=1}^{m} \sigma_k (A \cdot \cos(v_k \cdot t) + B \cdot \sin(v_k \cdot t))$$

The coefficients A and B are random variables that follow the standard normal distribution. The standard normal random function has two important properties that we will make use of to create our model, the mean and correlation. The estimation of the mean of the standard normal random function is shown below:

$$E[G(t)] = E\left[\sum_{k=1}^{m} \sigma_k (A \cdot \cos(v_k \cdot t) + B \cdot \sin(v_k \cdot t))\right]$$

By using linear properties of the expected value we can reduce the equation to the following:

$$\begin{aligned} E[G(t)] &= \textstyle\sum_{k=1}^{m} \sigma_k \cdot E[(A \cdot \cos(v_k \cdot t) + B \cdot \sin(v_k \cdot t))] \\ &= \textstyle\sum_{k=1}^{m} \sigma_k \cdot (E[A] \cdot \cos(v_k \cdot t) + E[B] \cdot \sin(v_k \cdot t)) \\ &= 0 \end{aligned}$$

The correlation between points at time t and time s can be estimated as follows,

$$\begin{aligned} \mathrm{Corr}[G(t), G(s)] &= E[G(t) \cdot G(s)] - E[G(t)] \cdot E[G(s)] \\ &= E\Big[\textstyle\sum_{kj=1}^{m} \sigma_k \cdot (A_k \cdot \cos(v_k \cdot t) + B_k \cdot \sin(v_k \cdot t)) \cdot \sigma_j \cdot (A_j + \cos(v_j \cdot s) + B_j \cdot \sin(v_j \cdot s))\Big] \\ &= E\Big[\textstyle\sum_{k,j=1}^{m} \sigma_k \sigma_j \cdot (A_k \cdot A_j \cdot \cos(v_k \cdot t) \cdot \cos(v_j \cdot s) + A_j \cdot B_k \cdot \sin(v_k \cdot t) \cdot \cos(v_j \cdot s) \ldots \\ &\quad A_k \cdot B_j \cdot \cos(v_k \cdot t) \cdot \sin(v_j \cdot s) + B_k \cdot B_j \cdot \sin(v_k \cdot t) \cdot \sin(v_j \cdot s)) \\ &= \sum_{k,j=1}^{m} \sigma_k \sigma_j \cdot (E[A_k + A_j] \cdot \cos(v_k \cdot t) \cdot \cos(v_j \cdot s) + E[A_j \cdot B_k] \cdot \sin(v_k \cdot t) \cdot \cos(v_j \cdot s) \ldots \\ &\quad E[A_k \cdot B_j] \cdot \cos(v_k \cdot t) \cdot \sin(v_j \cdot s) + E[B_k \cdot B_j] \cdot \sin(v_k \cdot t) \cdot \sin(v_j \cdot s)) \\ \mathrm{Corr}[G(t), G(s)] &= \sum_{k=1}^{m} \sigma_k^2 \cdot (\cos(v_k \cdot t) \cdot \cos(v_k \cdot s) + \sin(v_k \cdot t) \cdot \sin(v_k \cdot s)) \end{aligned}$$

By setting ζ equal to $s - t$, we can reduce the correlation to the following equation.

$$\mathrm{Corr}[G(t), G(s)] = C(\zeta) = \textstyle\sum_{k=1}^{m} \sigma_k^2 \cos(v_k \cdot \zeta)$$

Finally by setting t equal to s, or ζ equal to zero, the correlation function simplifies into the variance of the data set as follows:

$$\mathrm{Corr}[G(t), G(s)] = E[G(t) \cdot G(t)] = \mathrm{Var}[G(t)] = \textstyle\sum_{k=1}^{m} \sigma_k^2$$

This equation is a powerful tool for us because it allows us to build the correlation function, found in section 2.2.2., into our model by delineating values for ν_k and estimating values for σ_k. Values for ν_k were set to the following:

$$\begin{aligned} v_1 &= \frac{2\pi}{\tau} \\ v_k &= v_1 \cdot k \end{aligned}$$

Where τ is the length of dataset, 32.768 seconds. Expressing ν_k this way was chosen to allow an event related to ν_k to occur k times over the period of the dataset. Now that ν_k is defined, σ_k can be estimated using the following methodology.

$$C(\zeta) = \sum_{k=1}^{m} \sigma_k^2 \cdot \cos(v_k \cdot \zeta)$$

The correlation function shown above can be expanded into the equation below and $C(\zeta)$ can be set to a value $C(\zeta_1)$ from the correlation function found in section 2.2.2.

$$C(\zeta_1) = \sigma_1^2 \cos(v_1 \cdot \zeta_1) + \sigma_2^2 \cos(v_2 \cdot \zeta_1) + \ldots \sigma_m^2 \cos(v_m \cdot \zeta_1)$$

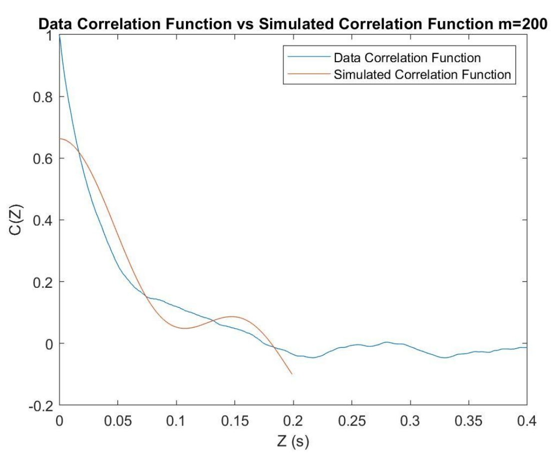
Next, this can be done with m values from the correlation function further expanding the equation above into a system of equations shown below:

$$\begin{aligned} &C(\zeta_1) = \sigma_1^2 \cos(v_1 \cdot \zeta_1) + \sigma_2^2 \cos(v_2 \cdot \zeta_1) + \ldots \sigma_m^2 \cos(v_m \cdot \zeta_1) \\ &C(\zeta_2) = \sigma_1^2 \cos(v_1 \cdot \zeta_2) + \sigma_2^2 \cos(v_2 \cdot \zeta_2) + \ldots \sigma_m^2 \cos(v_m \cdot \zeta_2) \\ &\ldots \\ &C(\zeta_m) = \sigma_1^2 \cos(v_1 \cdot \zeta_m) + \sigma_2^2 \cos(v_2 \cdot \zeta_m) + \ldots \sigma_m^2 \cos(v_m \cdot \zeta_m) \end{aligned}$$

Finally, this system of equations can be converted to a vector of known values (the correlation function found in 2.2.2.), a matrix of known coefficients ($cos(\nu_i \cdot \zeta_j)$), and a vector of unknowns ((σ_i^2)) — a solvable linear algebra problem

$$\overrightarrow{C(\zeta)} = \underline{A} * \vec{\sigma}^2$$

There is one complication with this approach. The issue lies in the fact that the values found for σ_i^2 must be greater than zero. The solution to this issue was to use least squares non-negative regression. The algorithm starts with a set of possible basis vectors and computes the associated dual vector lambda. It then selects the basis vector corresponding to the maximum value in lambda in order to swap out of the basis in exchange for another possible candidate. This continues until lambda ≤ 0 ("lsqnonneg"). Using this we were able to get reasonable estimates for σ_i^2. Below is a series of plots showing $\sigma_i^2 cos(\nu_i \cdot \zeta_j)$, shown in red, compared to the true correlation function shown in blue for $m = 170$ (left) and $m = 1000$ (right).



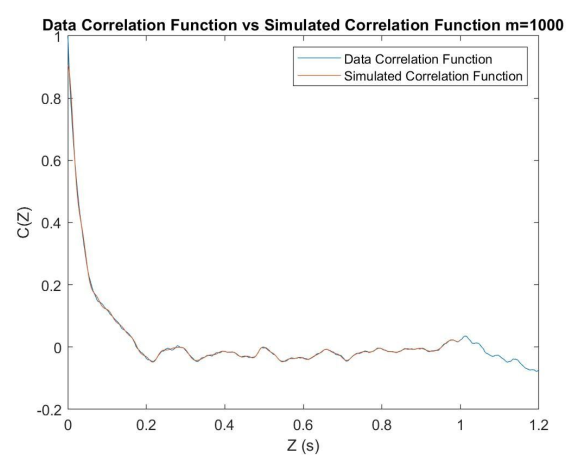


Figure 11. Graphs of the Data correlation function vs the simulated correlation function.

As shown in the plots estimates for σ_i^2 increase in accuracy with an increase in m. Additionally, when large value for m is used, the noise, as specified in section 2.2.2., is captured and increases the time for the regression process. To fix this, the function was truncated after the correlation function reaches zero and all values after were set to zero as shown in the plot below:

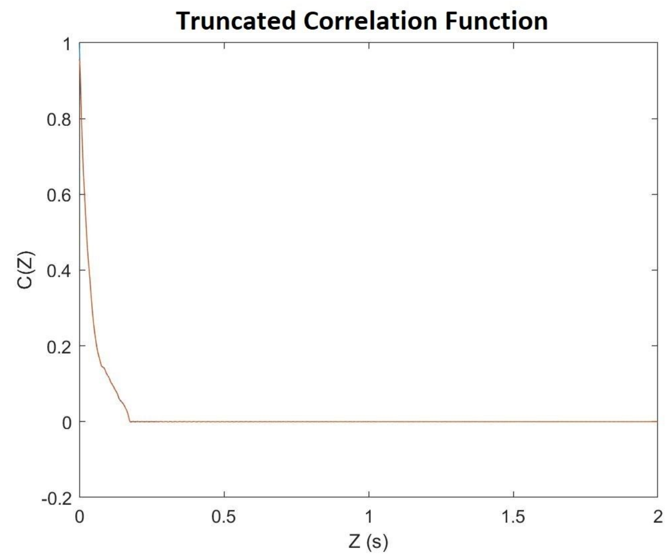


Figure 12. Graph showing the truncated correlation function where all values are set to 0 after the first negative value is reached.

Now that the standard normal random function $G(t)$ is defined, we can transform this function to the distribution found in section 3.1.1. by taking the normal cdf (Φ) of the function and applying the inverse of the GEV function.

$$X(t) = F^{-1}(\Phi(G(t))$$

$$X(t) = k\left(\frac{\ln(\Phi(G(t))-1}{\alpha}\right)^{-\alpha} + u$$

Finally, given the simulated function $X(t)$ we took the max value within the period for a large number of samples and found values for different exceedance probabilities.

4. Results

4.1. Data fitting

Throughout the building, some pressure taps had negative values for pressure coefficients that caused the MATLAB generated distributions to fit the data histograms poorly. To fix this problem, we fit distributions to the negative values of the dataset, and this gave much better results (Figure 13).

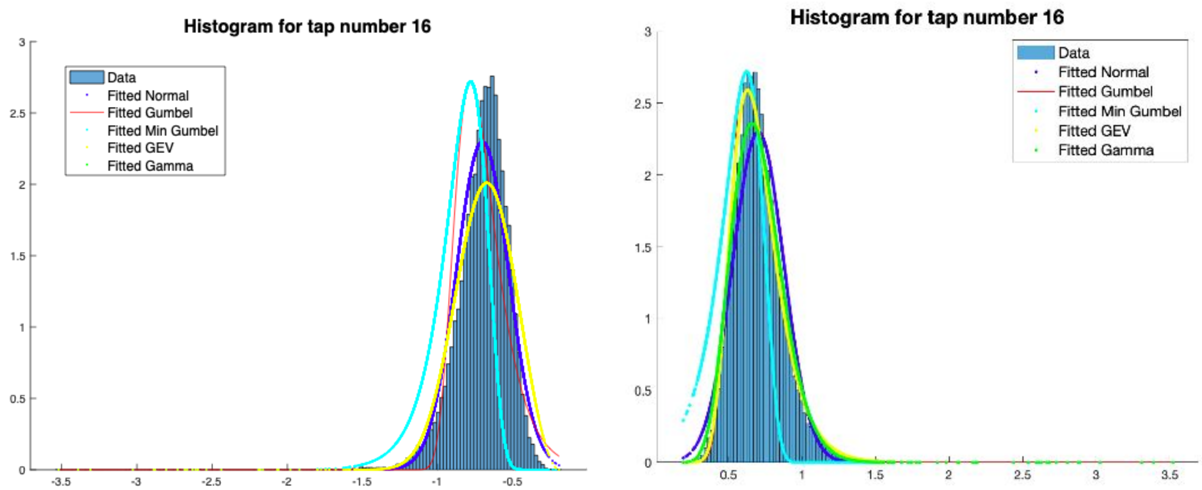


Figure 13. Example of a tap having negative values with poor distribution fits (left) and the improvement upon taking the negative values of the dataset for the same tap (right).

Data from pressure taps all over the building were plotted against different distributions listed in section 3.1, and Q-Q plots were created to determine which distribution fit the data the best (Figure 14). The GEV distribution was a near perfect fit for almost all the pressure taps on the building.

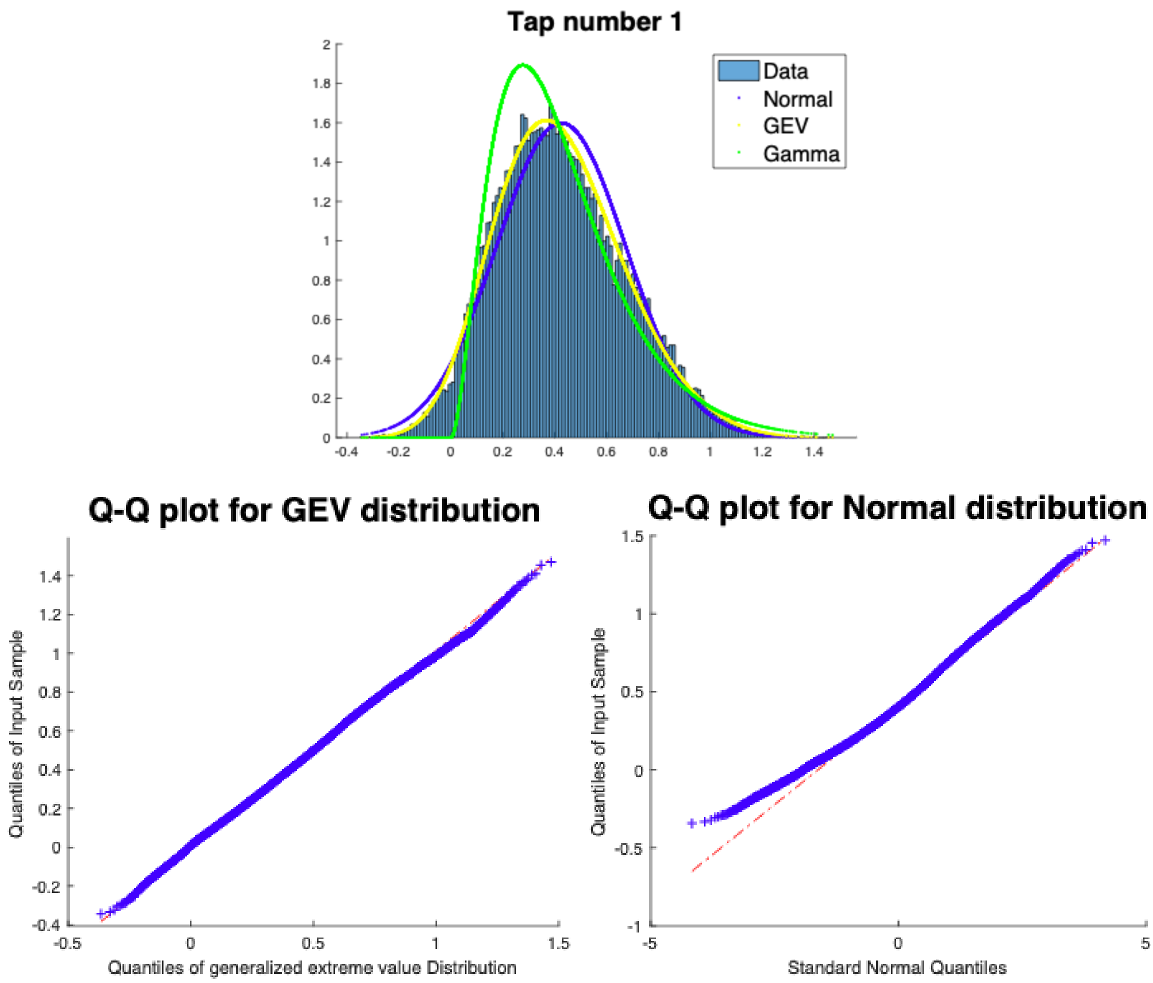
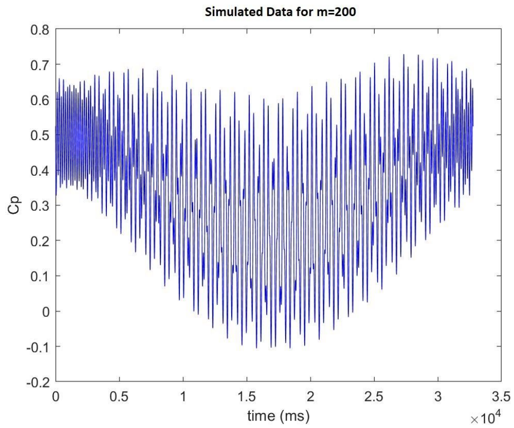


Figure 14. Graphs showing (clockwise) 1. histogram data and different distributions for Tap 1, 2. GEV Q-Q plot, 3) normal Q-Q plot. A Gamma Q-Q plot was not generated since the gamma distribution is clearly a poor fit for the data.

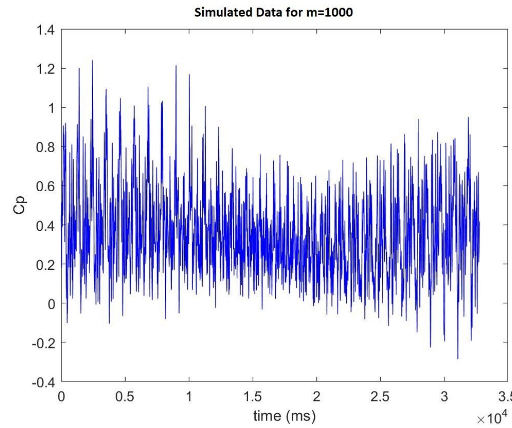
4.2. Generating Model and Samples

The plots below (Figure 15) show modeled pressure coefficient data generated for varying values of m, followed the true data for the tab being simulated.

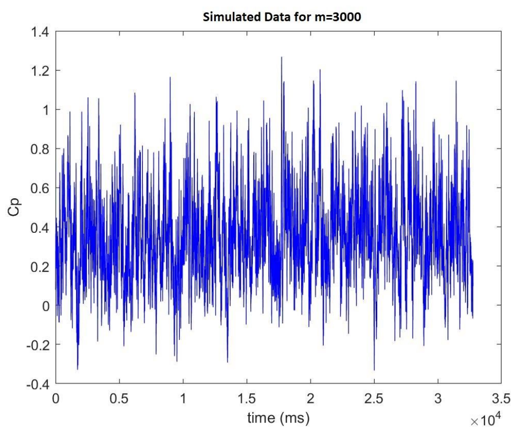
Simulated Data for m=200



Simulated Data for m = 1000



Simulated Data for m = 3000



Raw data for the tap being simulated

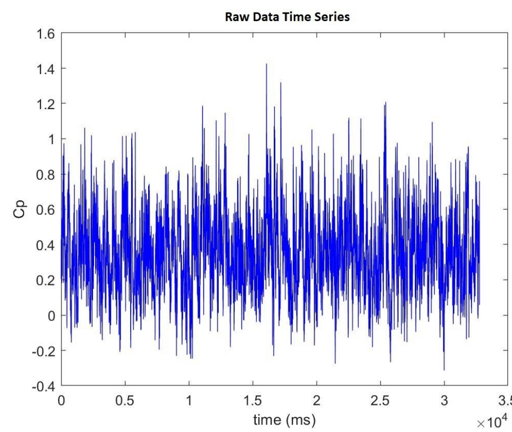
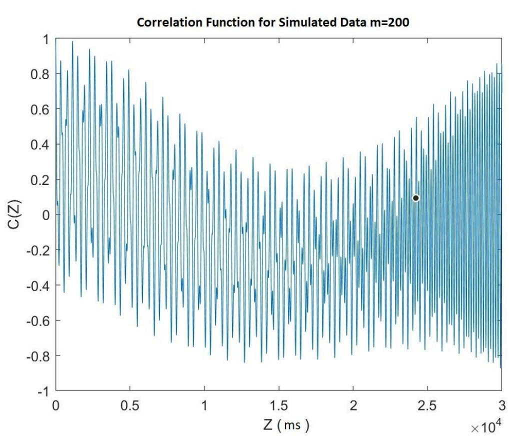
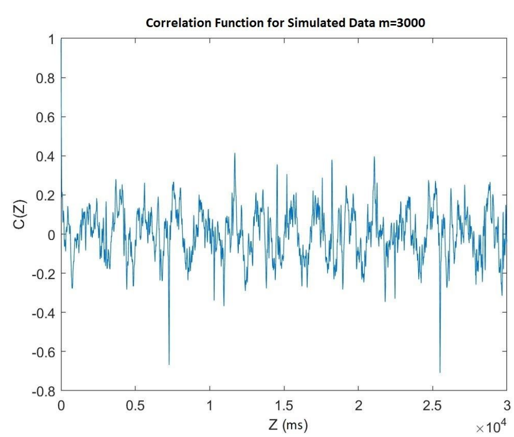
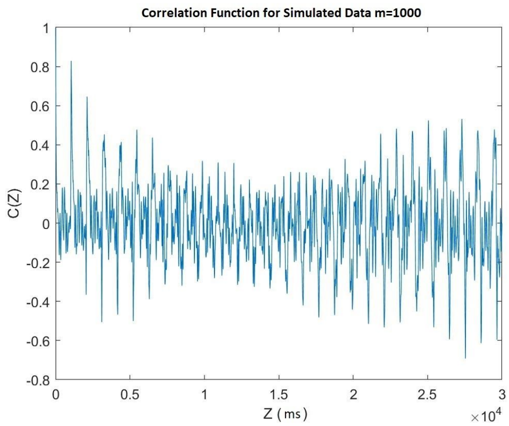


Figure 15. Pressure coefficients of 1) generated data for m = 200 (top-left), 2) generated data for m = 1000 (top-right), 3) generated data for m = 3000 (bottom-left), 4) raw data for the tap being simulated.

As shown by the plots, when m is low, certain frequencies dominate the modeled data. This behavior results in poor models that degrade the authenticity of the modeled data. The effect of dominant frequencies can be further investigated by plotting the correlation function as defined in section 2.2.2., shown below.







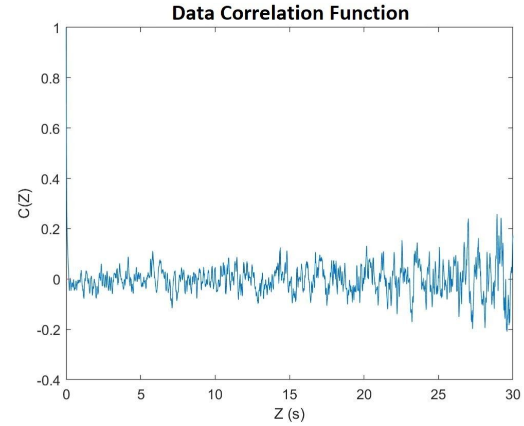


Figure 16. Correlation function of 1) generated data for m = 200 (top-left), 2) generated data for m = 1000 (top-right), 3) generated data for m = 3000 (bottom-left), 4) raw data for the tap being simulated.

Finally, to compare our model data to our raw data we would use a Q-Q plot of the model data versus the raw data. In the plots shown below, the Y data is the raw data and the X data is the modeled data.

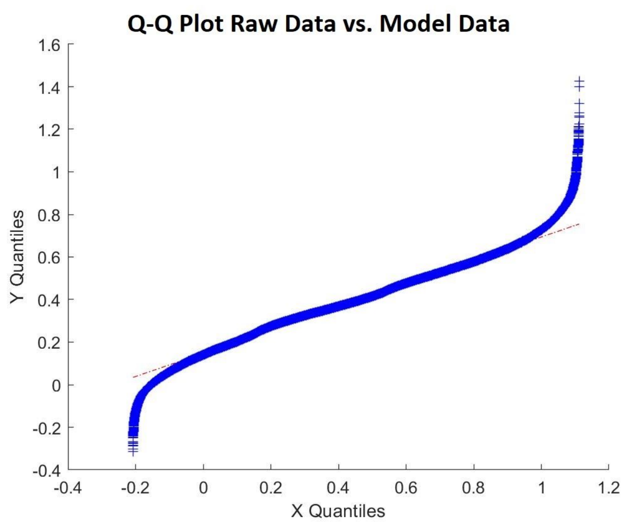


Figure 17. Q-Q plot of Raw data (Y axis) vs Model data (X axis) for m=100.

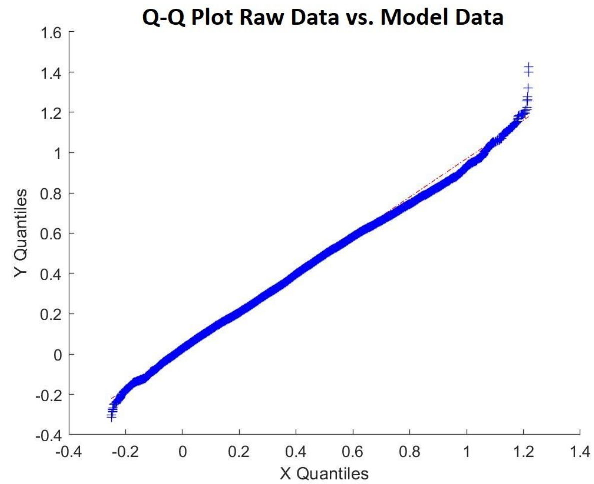


Figure 18. Q-Q plot of Raw data (Y axis) vs Model data (X axis) for m=1000.

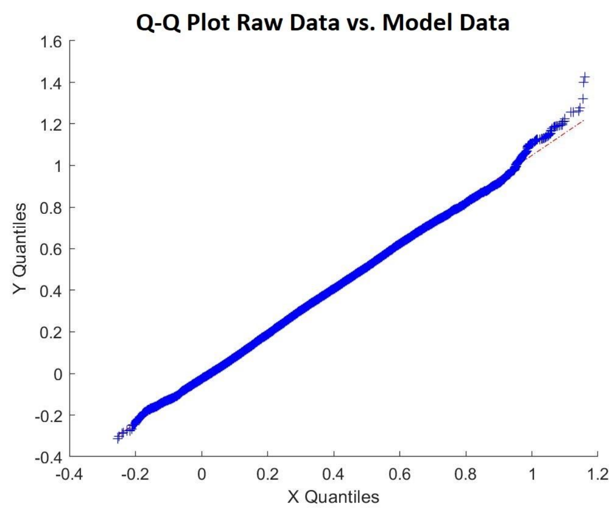


Figure 19. Q-Q plot of Raw data (Y axis) vs Model data (X axis) for m=4000.

Using our model we generated 1000 model data sets per tab with an m=100 and took the max value for each and determined the 0.05 exceedance value for each tab and compiled it into the following contour plot.

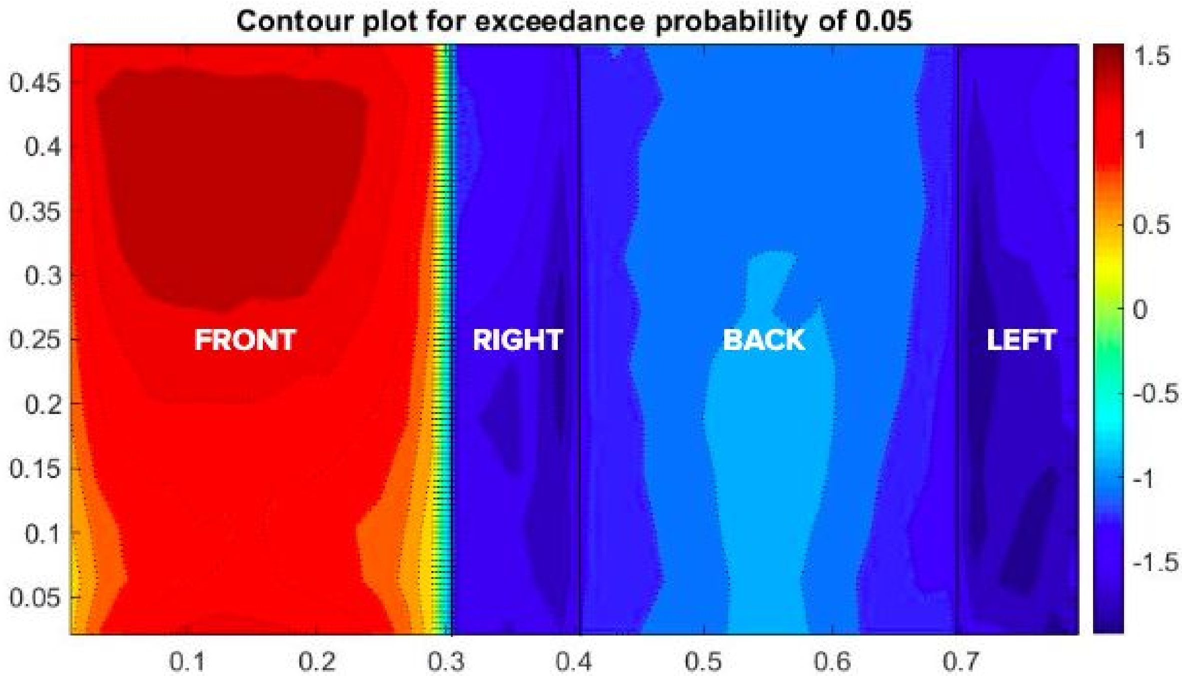


Figure 20. Contour plot of max values with an exceedance probability of 0.05 taken from models with an m=100.

5. Conclusions

5.1. Observations

The GEV distribution was necessary for our analysis due to its versatility in being able to capture skew and kurtosis of the raw data; something that could not be done by the normal or Gumbel distribution. The accuracy of our model was heavily dependent on the value m we used. The higher the m value was set, the more accurate our model data would become this can be seen in the correlation and QQ plots shown in observations.

As m increased the models correlation function neared that of the raw data's correlation function. This increase in accuracy can be attributed to the least squares non negative regression. As we increased m the σ^2 values could be estimated better through both increasing the number of frequencies involved, allowing some frequencies to be amplified, and eradicating interfering frequencies by setting some σ^2 values to zero. In fact a large proportion of the σ^2 values were set to zero, 82% when m was 3000. The Q-Q plots shown in observations reveal that the quantiles of the raw data and the model data get closer to converging with an increased m value. Even at a high m value of 3000 our model had trouble capturing the extreme values we are after and has a tendency to underestimate them. It is our hypothesis that as m is increased further we will be able to capture these values, as the plots show, with an increased m the model converges better. Our ultimate setback while running this analysis was the large amount of computations required to run the least squares non negative regression, a m equal to 3000 computation would take upwards of 15 minutes.

The contour plot shown in observations is there as a proof of concept, all of our analysis shows that an m value of 100 will not result in an accurate model. Additionally, this plot took over 30 minutes to generate.

5.2. Future Work

Future work for this project will be increasing the accuracy of our models then integrating those models into a contour plot. We believe that because a large proportion of the σ^2 values are set to zero, we may be able to find a pattern in the frequencies that can reduce the size of the least squares non negative regression computations. Additionally we would like to investigate spatial correlation between tabs.

6. References

"Lsqnonneg." *Lsqnonneg (MATLAB Functions)*, Northwestern University, www.ece.northwestern.edu/local-apps/matlabhelp/techdoc/ref/lsqnonneg.html.

Pinheiro, Eliane C., and Silvia L. P. Ferrari. "A Comparative Review of Generalizations of the Gumbel Extreme Value Distribution with an Application to Wind Speed Data." *Journal of Statistical Computation and Simulation*, vol. 86, no. 11, 2015, pp. 2241–2261., doi:10.1080/00949655.2015.1107909.

Simiu, Emil, and DongHun Yeo. *Wind Effects on Structures: Modern Structural Design for Wind.* Fourth edition., John Wiley & Sons, 2018, https://onlinelibrary.wiley.com/doi/book/10.1002/978111937589

Singh, V. P. *Entropy-Based Parameter Estimation in Hydrology*. Springer, 2011.

"Wind Effect on Buildings and Urban Environment - Aerodynamic Database of High-Rise Buildings." *Wind.arch.t*, Tokyo Polytechnic University, www.wind.arch.t-kougei.ac.jp/info_center/windpressure/highrise/Homepage/homepage

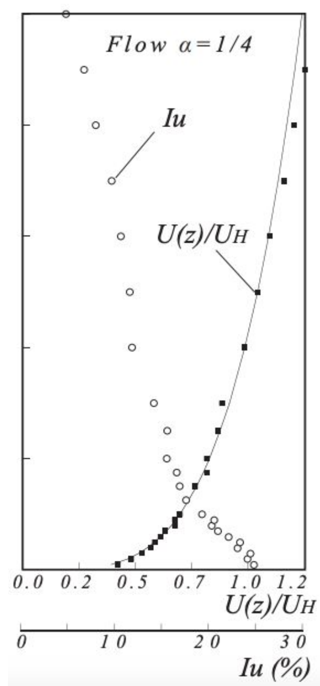
HDF.htm?fbclid=IwAR1WxczoDVZy0B6bJ9H7O8lEb0htQQlmc2NCy8ehyTQ0HUit7LfNyxfGt4E.

Appendix

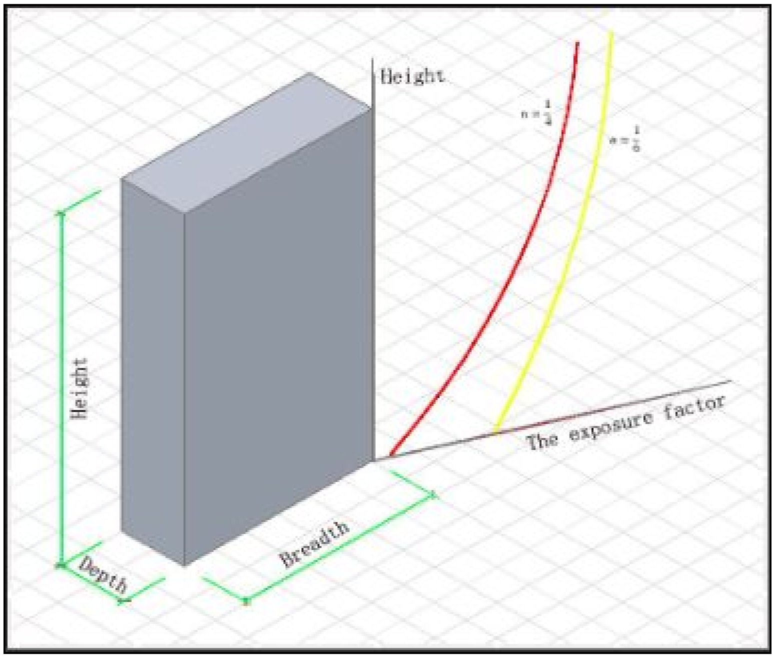
Supplementary data from TPU database

Source: Wind Pressure Database for High-Rise Building

Velocity profile in wind tunnel



Building profile



Distribution of pressure taps

Model geometrical parameters of a high-rise building: H=0.5m, B=0.3m, D=0.1m,Model scale=1/400.

MATLAB code written

Estimating moments of time series data

```
load('TPU_WindLoads_Data_Wide.mat');
dataset=Wind_pressure_coefficients;
mu = mean(dataset);
sig = std(dataset);
skew = skewness(dataset);
kurt = kurtosis(dataset);

Start=1; %Start tab of top row
END=40; %End tab of top row
W=40; %Total number of tabs in each row
Zmu = []; Zsig = []; Zskew = []; Zkurt = [];
X=[Location_of_measured_points(1,Start:END)];
Y=[];
for k=1:12
 Off=(k-1)*W;

Zmu=[Zmu;mu(Off+Start:Off+END)];
Zsig=[Zsig;sig(Off+Start:Off+END)];
Zskew=[Zskew;skew(Off+Start:Off+END)];
Zkurt=[Zkurt;kurt(Off+Start:Off+END)];
```

```
Y=[Y,Location_of_measured_points
(2,Off+Start)];
end
close all
figure
contourf(X,Y,Zmu,':') % Mean
caxis([min(mu) max(mu)])
colormap autumn
colorbar
daspect([1 1 1])
set(findall(gcf,'-property','Fon
tSize'),'FontSize',14)
title('Contour plot of
mean','FontSize',18)
figure
contourf(X,Y,Zsig,':') %
Standard deviation
caxis([min(sig) max(sig)])
colormap winter
colorbar
daspect([1 1 1])
set(findall(gcf,'-property','Fon
tSize'),'FontSize',14)
title('Contour plot of
SD','FontSize',18)
figure
contourf(X,Y,Zskew,':') % Skew
caxis([min(skew) max(skew)])
colormap spring
colorbar
daspect([1 1 1])
set(findall(gcf,'-property','Fon
tSize'),'FontSize',14)
title('Contour plot of
skewness','FontSize',18)
figure
contourf(X,Y,Zkurt,':') %
Kurtosis
caxis([min(kurt) max(kurt)])
colormap summer
colorbar
daspect([1 1 1])
set(findall(gcf,'-property','Fon
tSize'),'FontSize',14)
title('Contour plot of
kurtosis','FontSize',18)
```

Data-fitting

```
% Est_simplified.m
% Estimates mean, std, skewness,
kurtosis of a given dataset
where number
% of rows is number of samples
per pressure tap and number of
columns is
% the number of pressure taps
load('TPU_WindLoads_Data_Wide.ma
t')
dataset =
Wind_pressure_coefficients;
tap=1;
[ns,n] = size(dataset);
mu = mean(dataset);
sig = std(dataset);
skew = skewness(dataset);
kurt = kurtosis(dataset);

% Check how normal distribution
is
phi = cdf('normal',dataset(:,
tap),mu(tap),sig(tap));
fx = pdf('normal',
dataset(:,tap),mu(tap),sig(tap);
close all
figure
hold on
histogram(dataset(:,tap),'Normal
ization','pdf','NumBins',150) %
Plot density
plot(dataset(:,tap),fx,'.b') %
Plot normal distribution
title(['Tap number
',num2str(tap)],'FontSize',20)

%Check for GEV
p = gevfit(dataset(:,tap));
```

```
yGEV =
gevpdf(dataset(:,tap),p(1),p(2),
p(3));
plot(dataset(:,tap),yGEV,'.y')

%Check for gamma
k = mu(tap)^2/sig(tap)^2;
theta = sig(tap)^2/mu(tap);
yGamma =
pdf('gamma',dataset(:,tap),k,the
ta);
plot(dataset(:,tap),yGamma,'.g')
legend('Data','Normal','GEV','Ga
mma','FontSize',16)
hold off

%%%% BEGIN Q-Q PLOTS
%%normal
figure
qqplot(dataset(:,tap))
title('Q-Q plot for Normal
distribution','FontSize',20)
%%GEV
figure
qqplot(dataset(:,tap),fitdist(da
taset(:,tap),'GeneralizedExtreme
Value'))
title('Q-Q plot for GEV
distribution','FontSize',20)
%%Gamma
figure
pd=fitdist(dataset(:,tap),'Gamma
');
qqplot(dataset(:,tap),pd)
```

Creating model and generating samples

```
function [Values] =
Copy_of_Model(Data,m,ns,EXC,mu)
%{
Data - is the time series data
for a given tab
m - is this accuracy of the
random function
ns - is the number of samples
EXC - is the exceedance
probability we want to know
pressure coeff for
mu - is the mean of the time
series data for the given tab
Values - is the presure
coefficients for the given
exceedence probability
%}
Didwenegate=1;
if mu<0
 Data=-Data;
 Didwenegate=-1;
end

%Check for GEV
p = gevfit(Data);
yGEV =
gevpdf(Data,p(1),p(2),p(3));
%%%%%%%%%%%%%%%%%%%%% Generate C
plot
n=length(Data);
END=30000;
Count=0;
stopcount=1;

for k=1:END

C(k)=corr(Data(1:n-k),Data(k+1:n
));
 if C(k)>0
 Count=Count+1*stopcount;
 else
 stopcount=0;
 end
end

C=[1,C];
C(Count+1:END+1)=0;
Z=(0:1:Count+100)/1000;

%%%%%%%%%%%%%%%%%%%%% Guess at
sigmas
Z=0:0.001:0.001*(m-1);
T=32.768; % T=32.768;
Vk=((2*pi)/T)*(1:m);
```

```
VkZ=Vk'*Z;
COS=cos(VkZ);
SIGS=lsqnonneg(COS',C(1:m)');
SigKCos=SIGS(:,1).*COS;
Totalsigcos(:,1)=sum(SigKCos,1);
Err=sum((C(1:m)'-Totalsigcos).^2
,1); %not s=to sure how to
handle error
CGuess=sum(SIGS.*COS,1);

%%%%% GEN random normal time
function thing
t=0:0.001:T;
SIG=SIGS.^.5;
CT=cos(Vk'*t);
ST=sin(Vk'*t);
A=randn(m,ns); B=randn(m,ns);
% Generate Ak and Bk

for k=1:ns
G(:,k)=sum( SIG.*
(A(:,k).*CT+B(:,k).*ST));
end

for k=1:ns
model(:,k)=gevinv(normcdf(G(:,k)
,0,sum(SIGS)),p(1),p(2),p(3))';
Max(k)=max(model(:,k));
end

[f,x]=ecdf(Max);
IND = sum(f<(1-EXC));
Values =x(IND);
Values=Values*Didwenegate;
```