Predicting Turbulence Simulations

Enzo Moraes Mescall, Martin Olarte, Charlotte Coudert

2022-10-31

```
data.train = read.csv("data-train.csv")
data.test = read.csv("data-test.csv")
```

Introduction

Turbulence has been described as "the last great unsolved problem in classical physics" by renowned physicist Richard Feynman. It has not only been crucial in mathematics but has important applications in astrophysics, climatology, and engineering.

In this project, we explored the distribution and clustering of particles in an idealized turbulence (homogeneous isotropic turbulence or HIT) as affected by three independent predictors representing the properties of turbulent flow. Reynolds number (Re), Froude number (Fr), and Stokes number (St) quantify the fluid turbulence, gravitational acceleration, and particle characteristics, respectively. We used data from Direct Numerical Simulations (DNS) of Re, Fr, and St and their associated clusters produced, quantifying these distributions by their first 4 moments.

The purpose of this paper was twofold: scientific inference and predictive modeling. Our final models described below will help scientists understand how each parameter influences the probability distribution function (PDF) of particle cluster volumes while also being able to predict the PDF for a new parameter setting of (Re, F r, St), in terms of its four raw moments.

For moment 1, we were able to find a generalized linear model incorporating the interaction of Fr and St values as factors and log(Re). This model balances the trade-off of complexity and interpretability, as it gives us a moderate predictive accuracy while being interpretable in the context of turbulence with respect to the available parameters.

Methodology

The first noticeable thing about the data is that despite both Fr and Re being numerical predictors, they only have three levels each. Therefore, we decided to treat them as discrete predictors instead of interpolating or extrapolating values outside the given set for Re ($\{90, 224, 398\}$) and Fr ($\{0.052, 0.3, \infty\}$). This transformation was appropriate in the context of the problem since the testing data would also have values within the specified sets only. The caveat is that this analysis should not be directly used to understand other real-world problems like oceanic and atmospheric turbulent flow which have Re within the order of magnitude of 10^7.

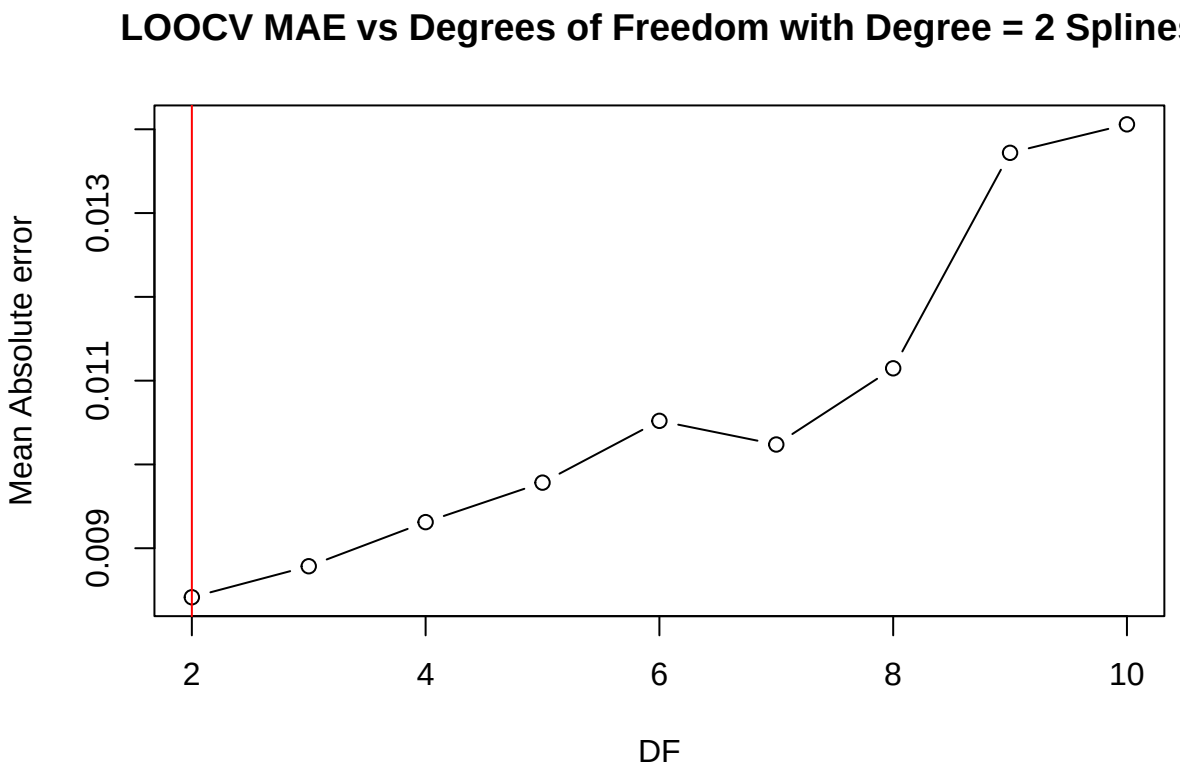
```
clean = data.train %>%
  mutate(Fr = as.factor(Fr),
         Re = as.factor(Re))
```

Since we have relatively few data points, n = 89, we decided to rely on LOOCV for our model selection process. We also vetted multiple different cost functions, from MSE to root MSE to root MSlogE to MAE.

The main reason for this was that MSE was over-inflating the negative of outliers and making some models seem as if they were performing worse than expected. Root MSE, or rMSE, and MAE helped to taper this by producing more interpretable values that were easier to visualize whilst still preserving the minimum and maximum of the MSE function. One step further was looking at root MSlogE, or RMSLE, which heavily scaled MSE and allowed us to compare the performance of one model to another across differently scaled response variables. For example if the 1st moment's model has an MSE of 0.1 and the 2nd moment's model has an MSE of 0.2, one would be inclined to assume the first model is superior, but since the values of the second moment tend to be orders of magnitude higher the latter model is actually performing a much more impressive estimation. This relative difference is much better captured in RMSLE.

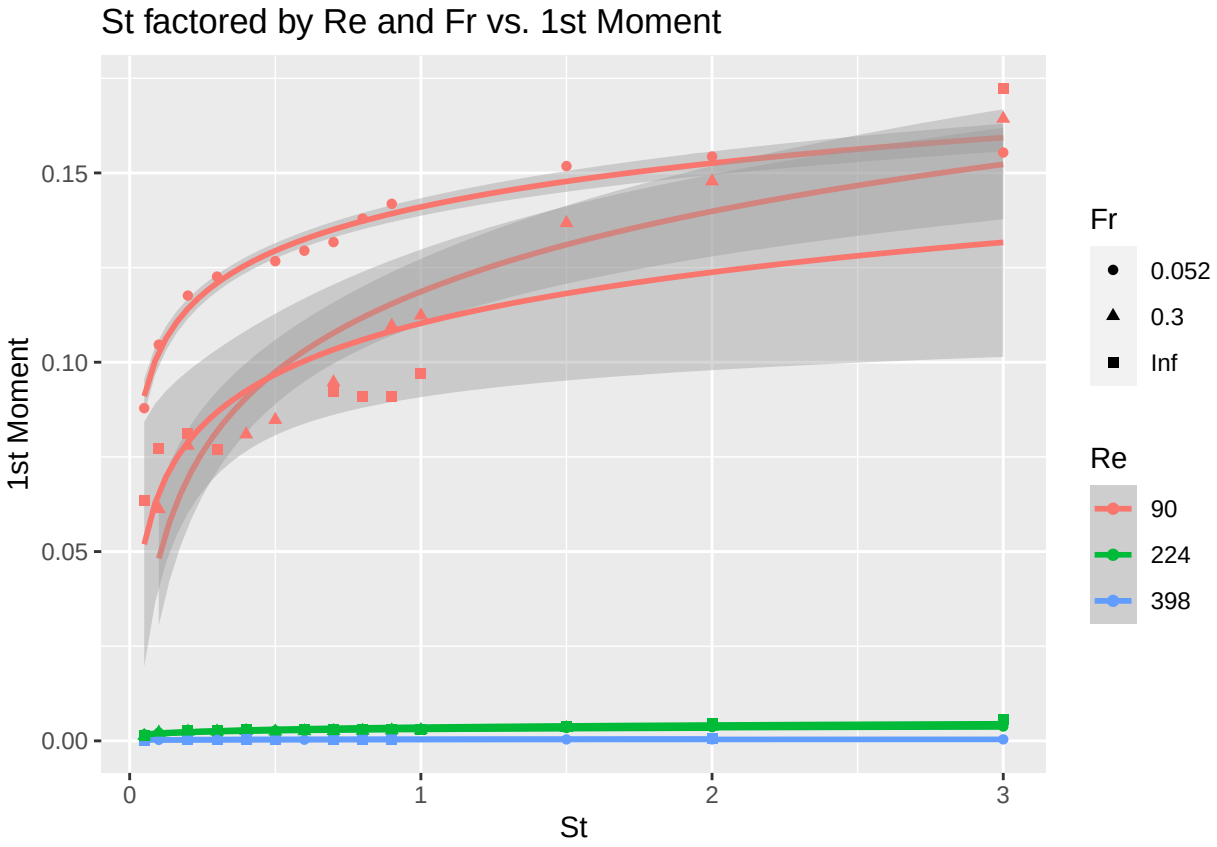
We tackled this problem sequentially, first focusing on models that provide information about the 1st moment, and moving forward until the 4th moment. To create a proficient model we estimated the test error performance of various different degrees of polynomials and splines with various combinations of degrees of freedom. An example is shown in the graph below of the process looking at the difference in LOOCV MAE between different numbers of knots for a quadratic spline model.

```
RMSLE = function(y_true, y_pred) sqrt(mean((log(y_true + 1) - log(y_pred + 1))^2))
MAE = function(y_true, y_pred) mean(abs(y_true - y_pred))

errors = rep(NA, 10)
for (i in 2:10) {
  lm = glm(R_moment_1 ~ bs(St, df = i, degree = 2)*Re + Fr, data = clean)
  errors[i] = suppressWarnings(cv.glm(clean, lm, cost = MAE)$delta[1])
}
plot(2:10, errors[-1], type="b", xlab="DF", ylab="Mean Absolute error",
     main = 'LOOCV MAE vs Degrees of Freedom with Degree = 2 Splines')
abline(v = which.min(errors), col = "red")
```

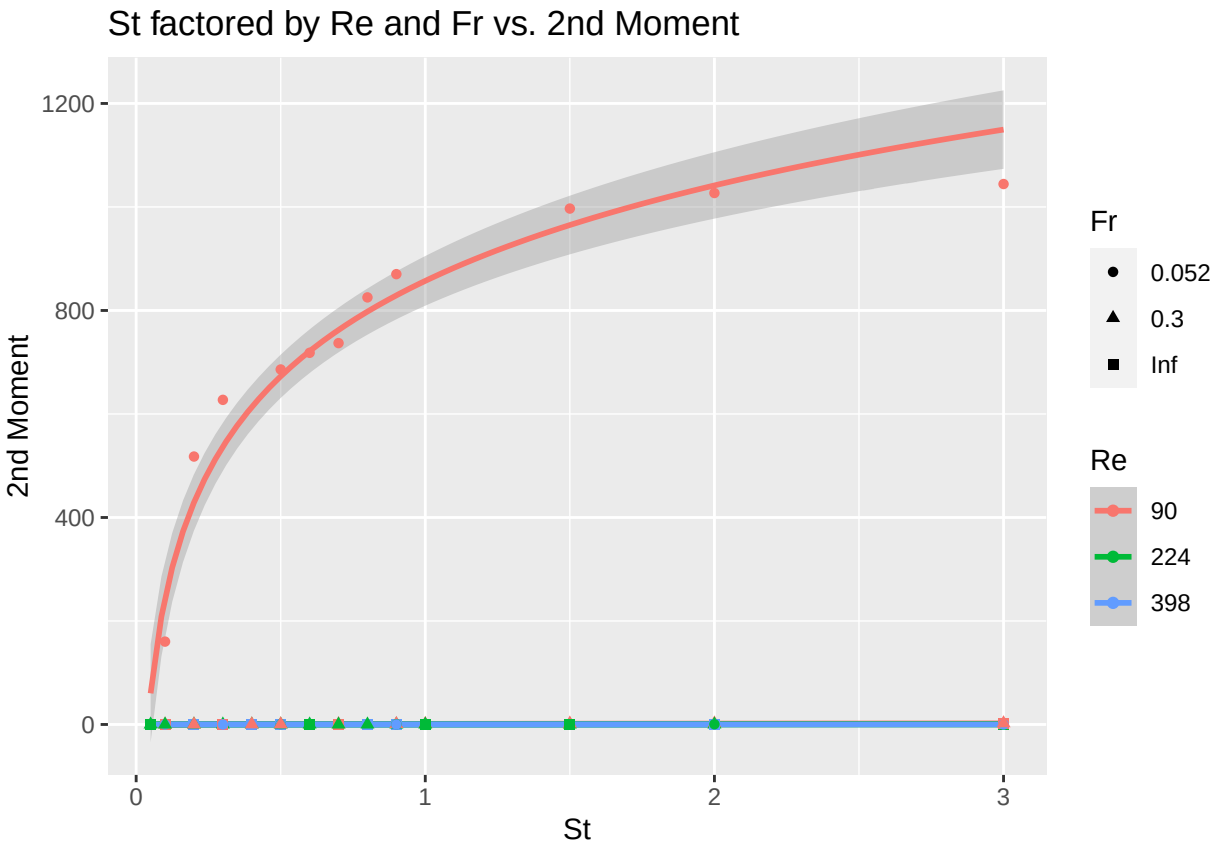


Notably, the simplest model has won out. Then, the next step was to look at what kind of interactions we may need. To identify these interactions, we plotted the data and performed some exploratory data analysis.

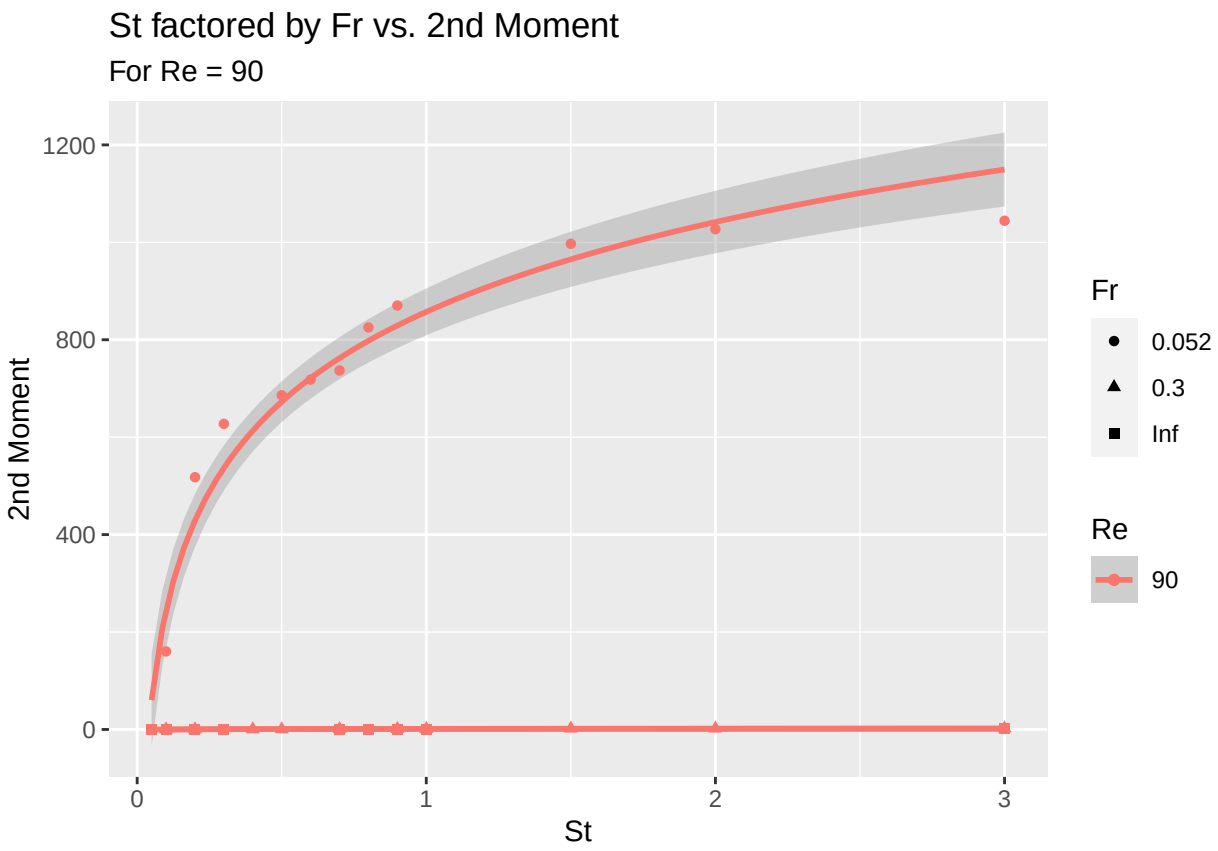
```
re_over = data.train %>%
  filter(Re == 90) %>%
  mutate(Fr = as.factor(Fr),
         Re = as.factor(Re))

re_under = data.train %>%
  filter(Re != 90) %>%
  mutate(Fr = as.factor(Fr),
         Re = as.factor(Re))


par(mfrow=c(2,2))

  ggplot(aes(x = St, y = R_moment_1, color = Re, shape = Fr), data = clean) +
  labs(title = "St factored by Re and Fr vs. 1st Moment", x = "St", y = "1st Moment") +
  geom_smooth(method = "lm", formula = "y ~ log(x)") +
  geom_point()
```



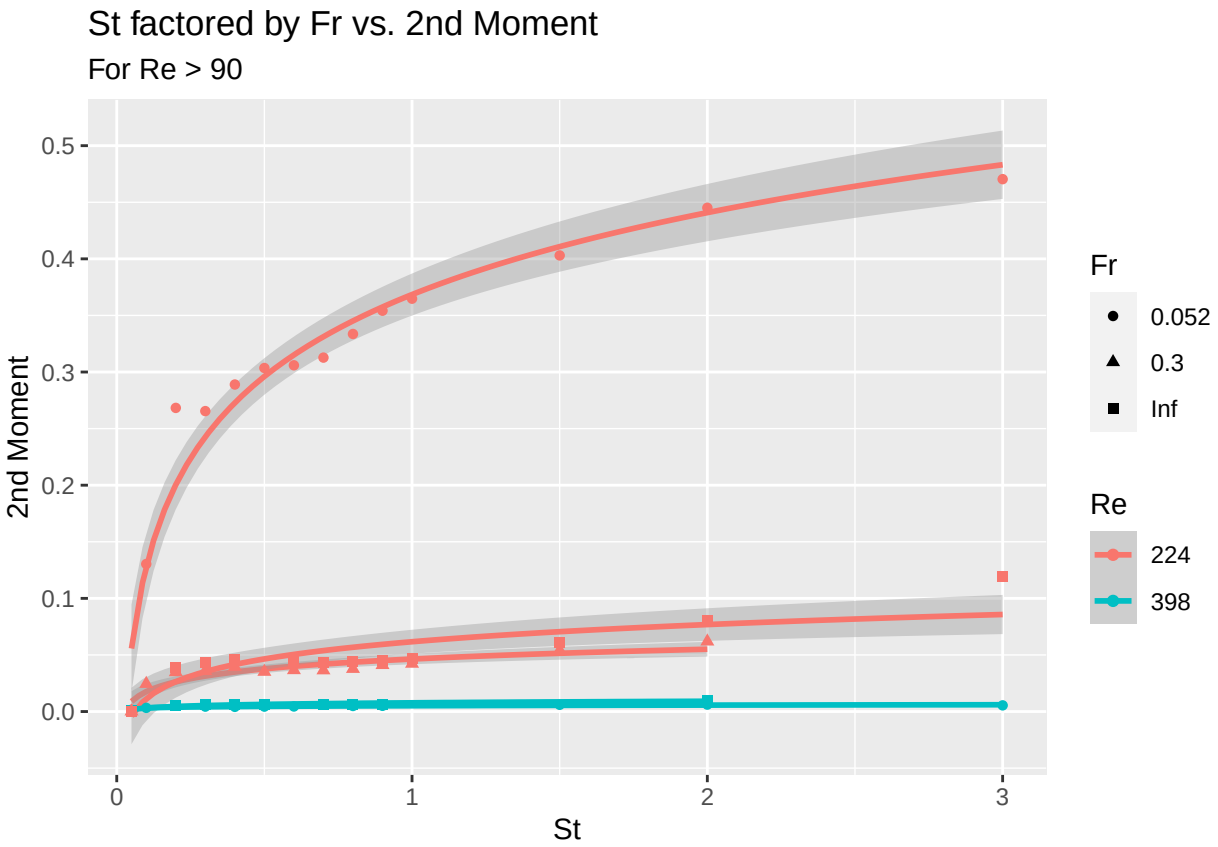

```
ggplot(aes(x = St, y = R_moment_2, color = Re, shape = Fr), data = clean) +
labs(title = "St factored by Re and Fr vs. 2nd Moment", x = "St", y = "2nd Moment") +
geom_smooth(method = "lm", formula = "y ~ log(x)") +
geom_point()
```




```
ggplot(aes(x = St, y = R_moment_2, color = Re, shape = Fr), data = re_over) +
labs(title = "St factored by Fr vs. 2nd Moment", subtitle = "For Re = 90", x = "St", y = "2nd Moment")
geom_smooth(method = "lm", formula = "y ~ log(x)") +
geom_point()
```




```
ggplot(aes(x = St, y = R_moment_2, color = Re, shape = Fr), data = re_under) +
labs(title = "St factored by Fr vs. 2nd Moment", subtitle = "For Re > 90", x = "St", y = "2nd Moment")
  geom_smooth(method = "lm", formula = "y ~ log(x)") +
  geom_point()
```



As we get to further moments, it becomes very clear that the majority of our error came from the much larger values all associated with an Re of 90 and an Fr of 0.052. These looked to follow a logarithmic path of St and when we applied that to all the interaction terms, we found our best model yet.

Our model selection process began with a very complex model containing a complete set of interactions terms, which included triple interaction terms across the three predictors (and abiding by the strong hierarchy principle all component interactions and main effects), and included log, square root, and inverse terms. We then performed backwards model selection using LOOCV, MSE, and MAE to select optimal predictors and all metrics agreed on the following models for both the first and second moments.

```
model_m1 = glm(R_moment_1 ~ log(St)*Fr*Re + St*Fr*Re, data = clean)
model_m2 = glm(R_moment_2 ~ log(St)*Fr*Re + St*Fr*Re, data = clean)
```

After this, we created a function that would resemble our final product and take in a tuple of St, Re, and Fr and then return the four predicted moments. We reserved the global variable names `model_mX` for this final function.

```
# temp models
predictive_model = function(test_St, test_Re, test_Fr) {
  newdata = data.frame(list(St = test_St,
                            Re = as.factor(test_Re),
                            Fr = as.factor(test_Fr)))

  m1 = predict(model_m1, newdata =  newdata)
  m2 = predict(model_m2, newdata =  newdata)
  m3 = predict(model_m3, newdata =  newdata)
  m4 = predict(model_m4, newdata =  newdata)
```

```
  return(c(m1, m2, m3, m4))
}
```

Results

Conclusion