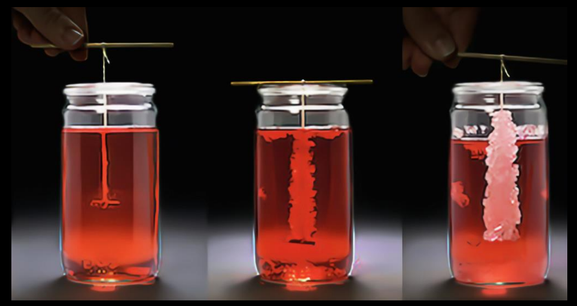
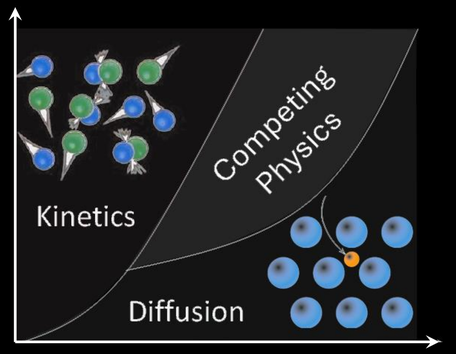
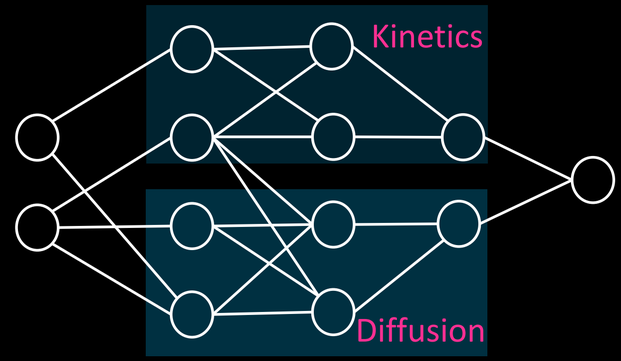
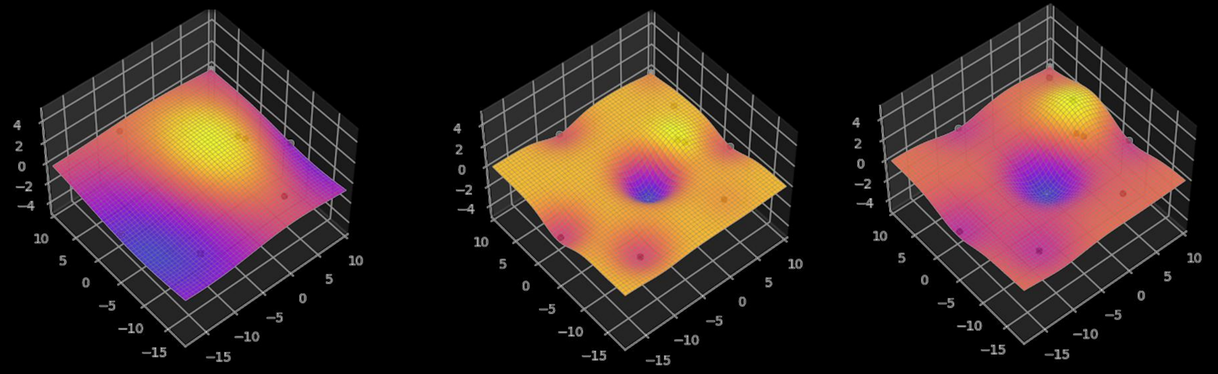
Physics Informed Machine Learning Adhesive Bonding







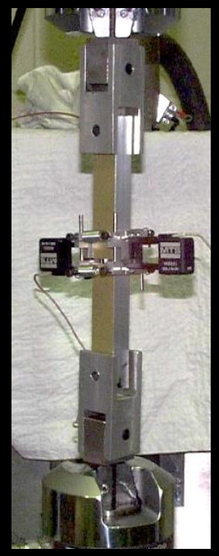


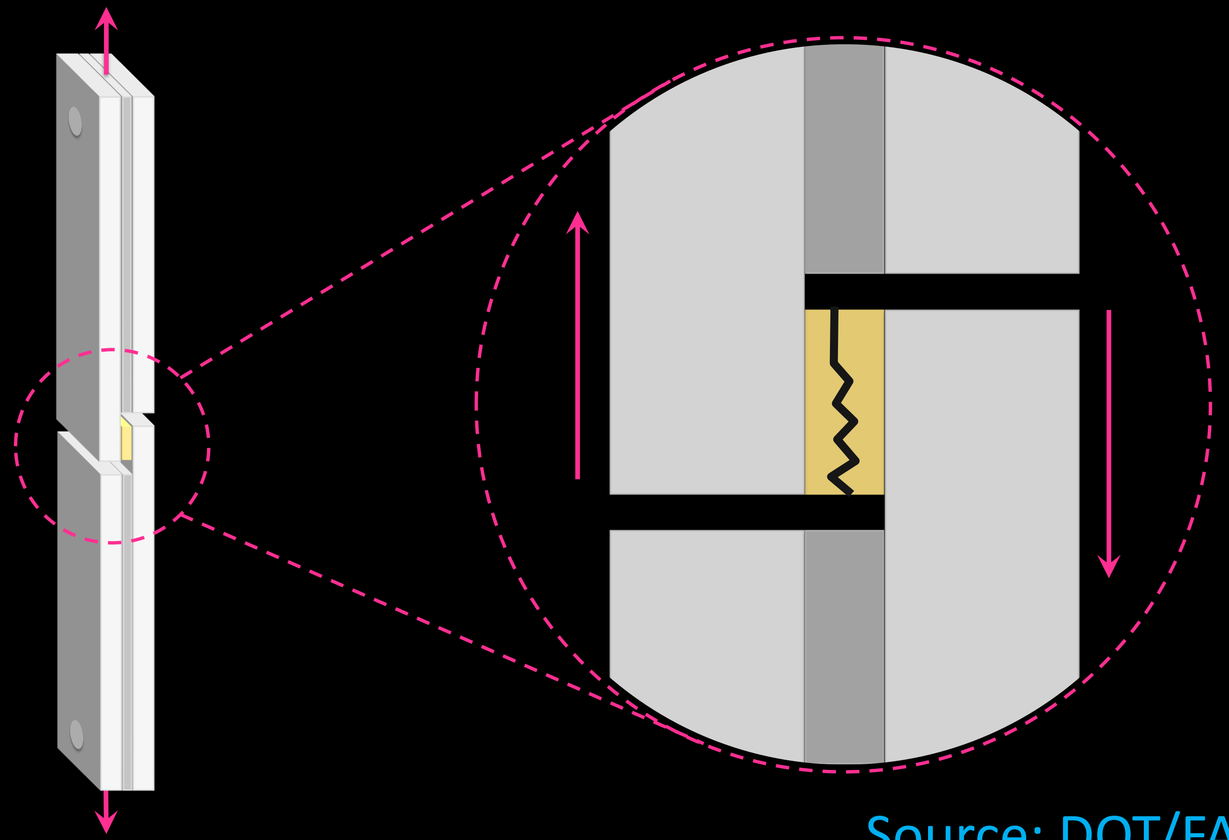
Navid Zobeiry, Associate Professor
Email: navidz@uw.edu
Materials Science & Engineering
https://composites.uw.edu/AI/

UNIVERSITY of WASHINGTON

Background

- Adhesive bonding tests were conducted on MGS A100/B100 per ASTM D5656, using aluminum substrates.

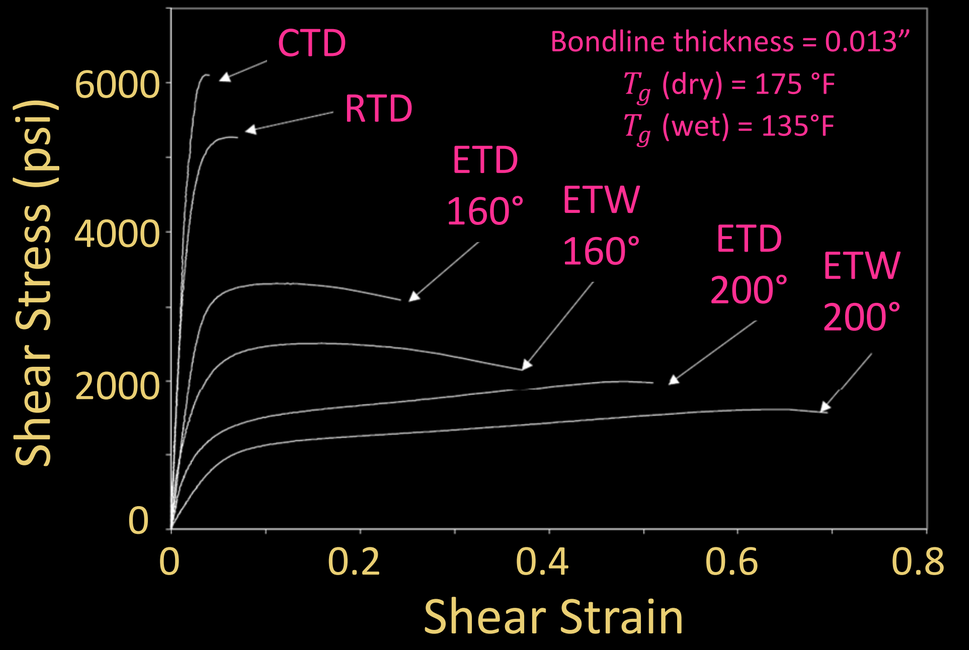




Source: DOT/FAA/AR-01/33

Background

- **23 shear tests are conducted at various humidity levels and temperatures:**
  - **CTD**: -65°F, 0% RH
  - **RTD**: 70°F, 0% RH
  - **ETD 160°**: 160°F, 0% RH
  - **ETW 160°**: 160°F, 85% RH
  - **ETD 200°**: 200°F, 0% RH
  - **ETW 200°**: 200°F, 85% RH



Source: DOT/FAA/AR-01/33

Dataset

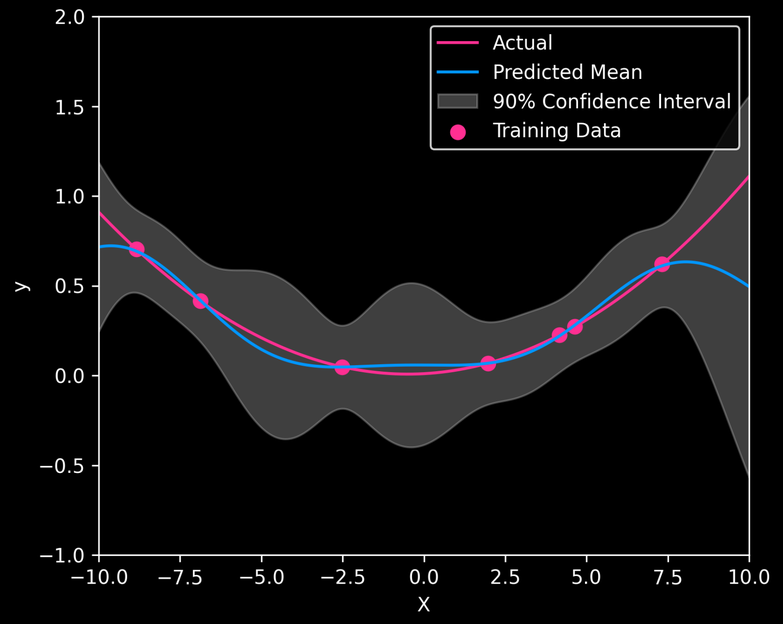
adhesive/adhesive.csv

ID	Sample#	Type	RH%	T (F)	Strength (psi)
1	C11A22	Test	0	-65	6903.1
2	C11A31	Test	0	-65	6645.2
3	C11A43	Test	0	-65	6467
4	C11A55	Test	0	-65	6109.2
5	C11A74	Test	0	-65	5831.8
6	C11A53	Train	0	70	5062
7	C11A56	Train	0	70	5281
8	C11A61	Train	0	70	4835.6
9	C11A24	Train	0	160	3306.1
10	C11A34	Train	0	160	3248.8
11	C11A52	Train	0	160	3642.6
12	C11A23	Train	85	160	2509.4
13	C11A32	Train	85	160	2541.4
14	C11A72	Train	85	160	2307.4
15	C11A76	Train	85	160	2324.9
16	C11A26	Train	0	200	1872.4
17	C11A36	Train	0	200	1996.9
18	C11A45	Train	0	200	2116
19	C11A51	Train	0	200	2040.1
20	C11A54	Train	0	200	2349.2
21	C11A25	Test	85	200	1590.9
22	C11A35	Test	85	200	1683.7
23	C11A62	Test	85	200	1365.1

Source: DOT/FAA/AR-01/33

Problem Statement

- Train a Gaussian Process Regression (GPR) model to predict shear strength as a function of temperature and humidity:
- Training Data:
  - **RTD**: 70°F, 0% RH
  - **ETD 160°**: 160°F, 0% RH
  - **ETW 160°**: 160°F, 85% RH
  - **ETD 200°**: 200°F, 0% RH
- Test Data:
  - **CTD**: -65°F, 0% RH
  - **ETW 200°**: 200°F, 85% RH



Python Implementation

> Import GPR libraries

```
from sklearn.gaussian_process import GaussianProcessRegressor
from sklearn.gaussian_process.kernels import RBF, DotProduct, Matern
from sklearn.gaussian_process.kernels import ConstantKernel as C, WhiteKernel as W
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import StandardScaler
```

> Load and prepare data

```
data_file = 'adhesive.csv'
df = pd.read_csv(data_file)
```

```
# Features
X_columns = ['T_F', 'RH']
```

```
# Output
Y_columns = ['strength_psi']
```

```
X = df[X_columns]
Y = df[Y_columns].values.ravel()

# Split the data based on the 'Type' column
X_train = X[df['Type'] == 'Train']
X_test = X[df['Type'] == 'Test']
Y_train = Y[df['Type'] == 'Train']
Y_test = Y[df['Type'] == 'Test']

# Scale the data
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
X_scaled = scaler.transform(X)
```

Python Implementation

> Iterate over a list of kernels to identify the optimal one:

```
# GPR Kernels to try
kernels = [
    ("RBF+noise", RBF() + W()),
    ("Dot+noise", DotProduct() + W()),
    ("Matern+noise", Matern() + W()),
    ("Dot^2+noise", DotProduct()**2 + W()),
    ("RBF+Dot+noise", RBF() + DotProduct() + W()),
    ("Matern+Dot+noise", Matern() + DotProduct() + W()),
]
```

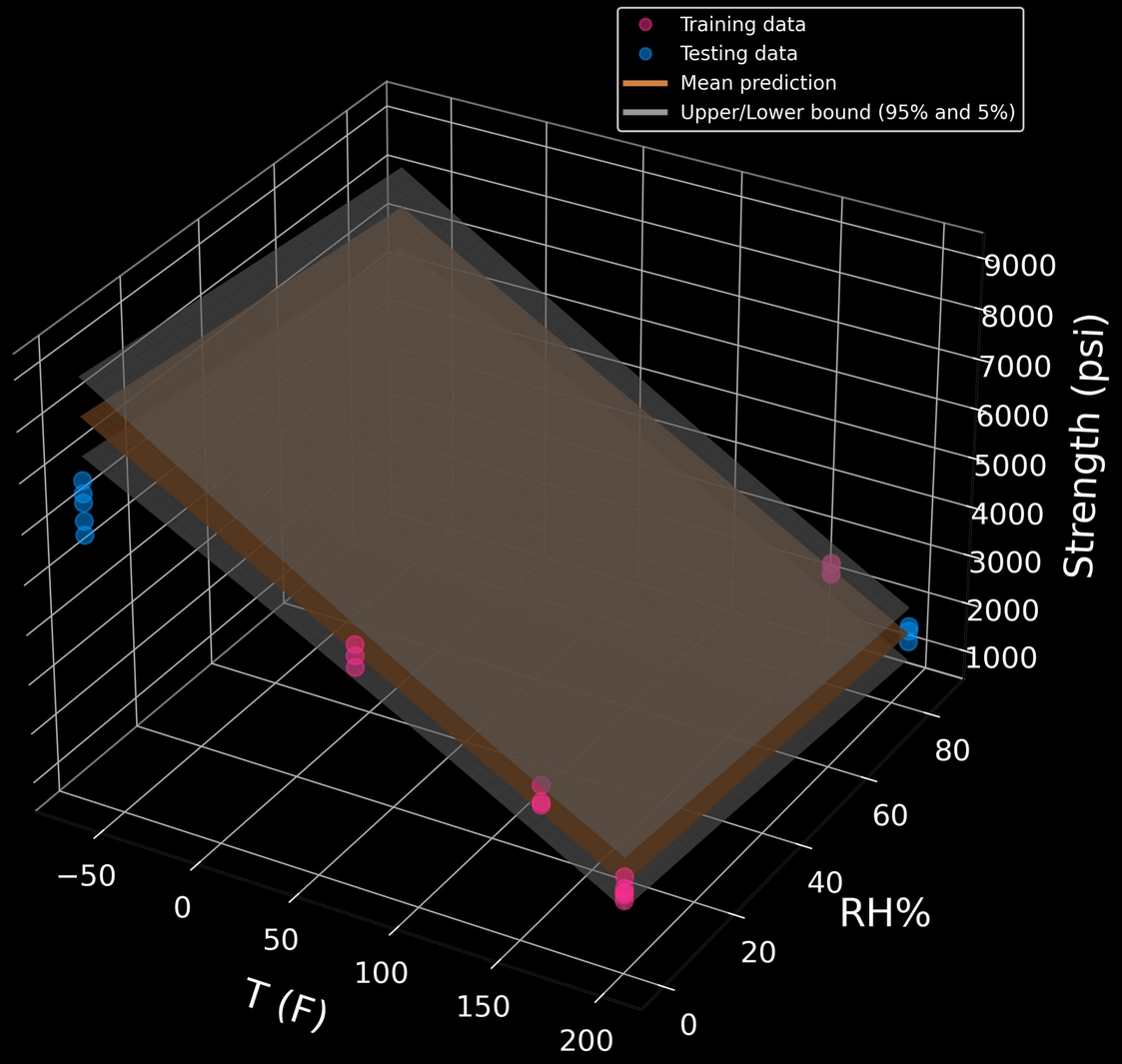
```
for name, kernel in kernels:
    gpr = GaussianProcessRegressor(kernel=kernel, alpha=1e-4, n_restarts_optimizer=50, random_state=42)
    try:
        gpr.fit(X_train, Y_train)

        # Predict on training and testing data
        Y_pred_train, std_train = gpr.predict(X_train, return_std=True)
        Y_pred_test, std_test = gpr.predict(X_test, return_std=True)

        # Calculate RMSE for training and testing data
        rmse_train = round(np.sqrt(mean_squared_error(Y_train, Y_pred_train)))
        rmse_test = round(np.sqrt(mean_squared_error(Y_test, Y_pred_test)))

        results.append({"Model": name, "RMSE_test": rmse_test, "RMSE_train": rmse_train})
```

```
mean_prediction, std_prediction = best_model.predict(X_scaled, return_std=True)
df['predicted_strength'] = mean_prediction
df['upper_bound'] = mean_prediction + 1.96 * std_prediction  # 95% confidence interval
df['lower_bound'] = mean_prediction - 1.96 * std_prediction  # 5% confidence interval
```

Traditional ML: Training and Evaluation

- **Best Kernel:**
  - **Dot^2 + noise**
  - **RMSE test: 1423 psi**
  - **RMSE train: 219 psi**

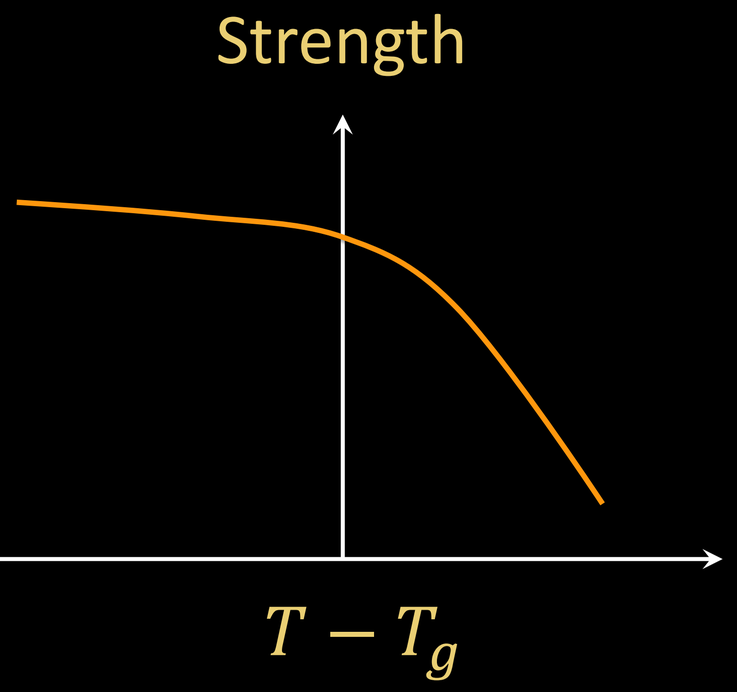
Kernel	RMSE test	RMSE train
Dot^2+noise	1423	219
Dot+noise	6827	1130
RBF+Dot+noise	2827	1130
Matern+Dot+noise	2827	1130
RBF+noise	5150	3233
Matern+noise	5150	3233



adhesive/adhesive.py

Underlying Physics

- Polymer strength decreases as temperature nears the glass transition, T_g.
- A sharp drop occurs at or above T_g.
- T_g for our material:
  - 175 °F (dry, RH 0%)
  - 135°F (wet, RH 85%)



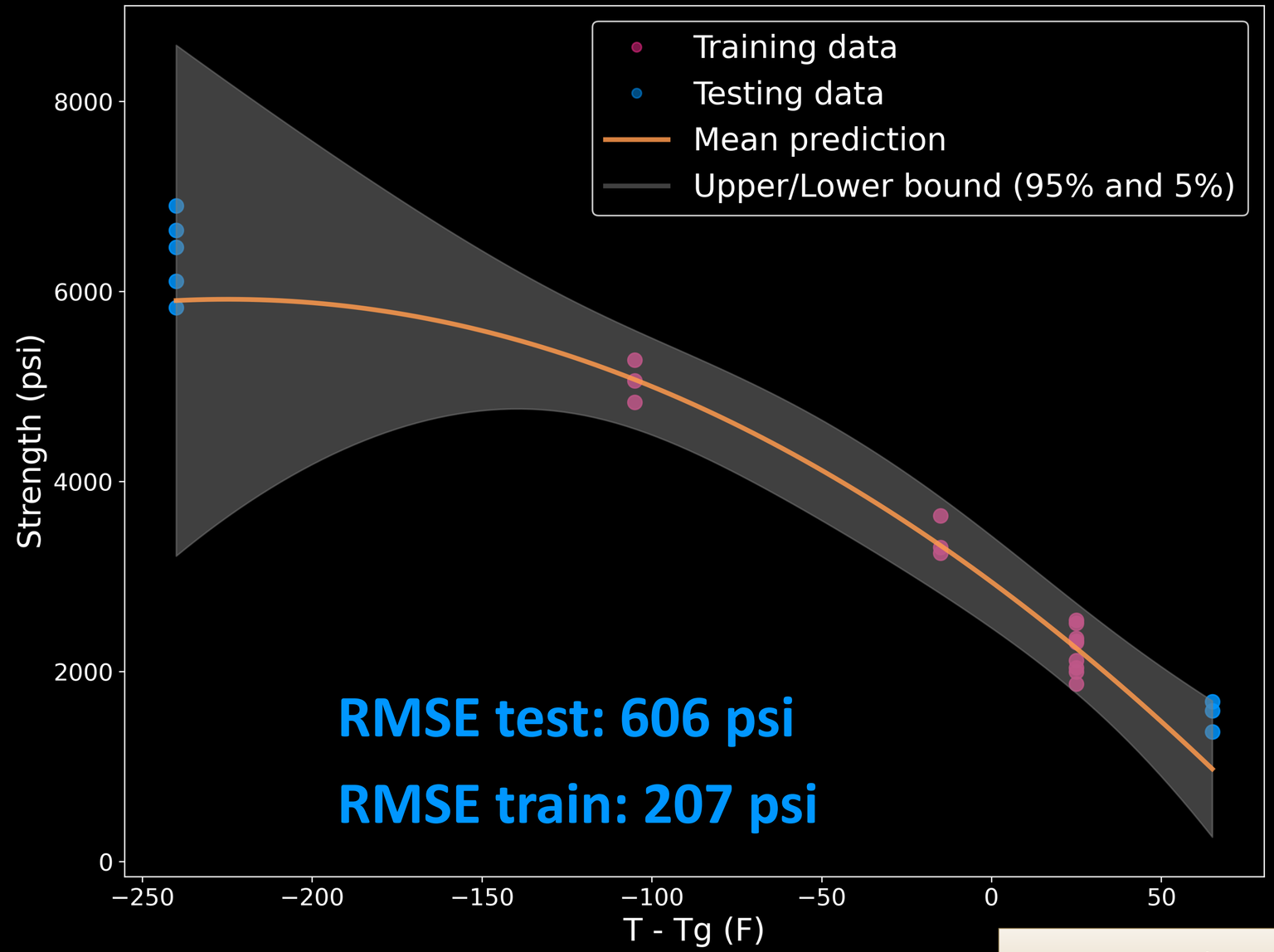
Physics-informed Machine Learning

> Code Changes:

- Introduce $T - T_g$ as a new parameter.
- Use a second-order polynomial kernel to capture strength versus $T - T_g$.
- Simplify input to only $T - T_g$ for strength prediction.

```
if option == 2:
    # physics-informed feature
    X_columns = ['T_Tg']
    df['T_Tg'] = df['T_F'] - df['Tg_F']

    #physics-informed GPR Kernel
    kernels = [
        ("Dot^2+Dot+C+noise", C() * DotProduct()**2 + C() * DotProduct() + C() + W()),
    ]

    # Output
    Y_columns = ['strength_psi']
```

Physics-informed Machine Learning



adhesive/adhesive.py