Machine Learning With Functional Data

Chris Gotwalt
Director of JMP Statistical R&D
JMP Division, SAS Institute

INTRODUCTION

- A lot of data is really *functional* in form:
  - Time series data
  - Spectra
  - Measurements taken over a range of temperatures, distances, etc.
  - Sensor streams from manufacturing processes, etc.

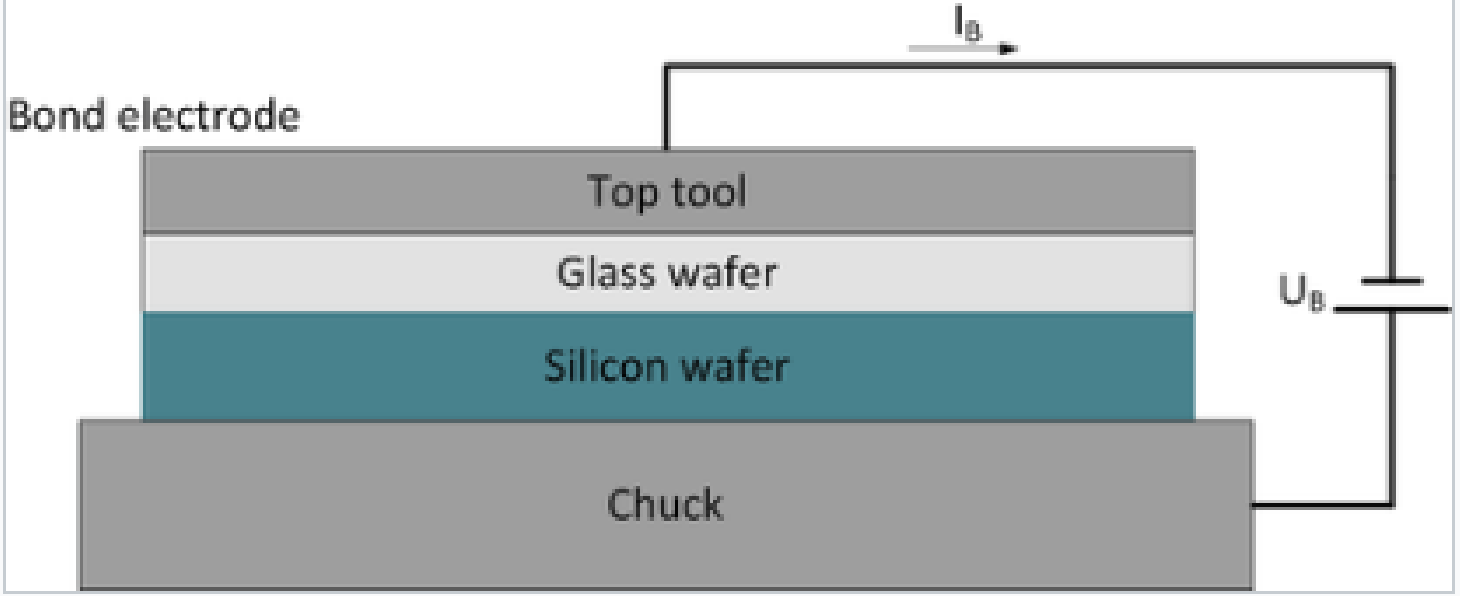
- Sensors create data with particular challenges
  - Generates lots of measurements: redundant information / high autocorrelation
  - Possibly irregular sampling times

- How can you use this data to predict something?
  - Try to use all the data?
  - Try to extract relevant features?

ANODIC BOND DATA

- Data are from a step in a semiconductor manufacturing process where there is a step where glass is bonded to the devices (anodic bonding).
- Data shown are simulated using models fit to the real data
- The bonding step is problematic: destroys about 11% of the wafers.
- The wafers cannot be tested for several weeks, after the rest of the production process steps are applied.
- We want to find a machine learning algorithm/model that will correctly predict which units are bad shortly after the anodic bonding step.

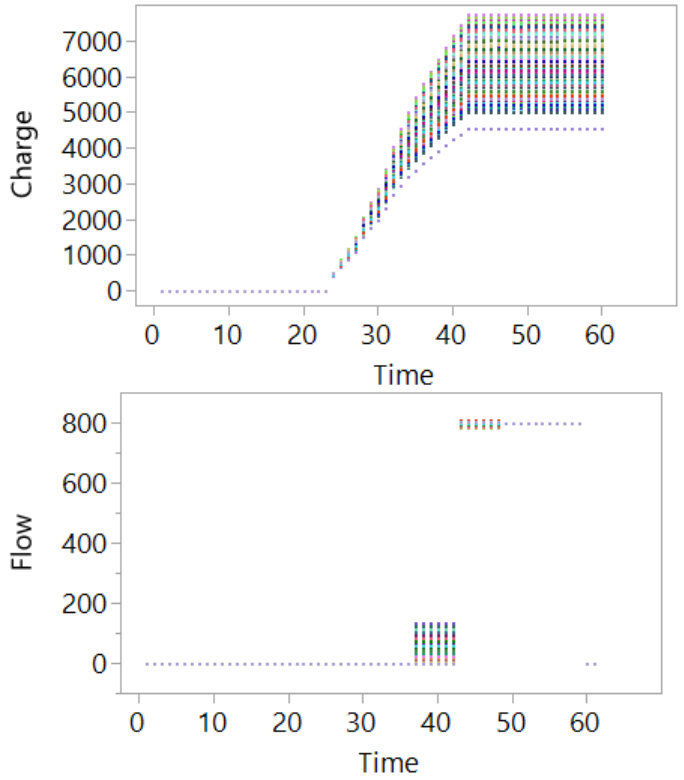
ANODIC BOND DATA

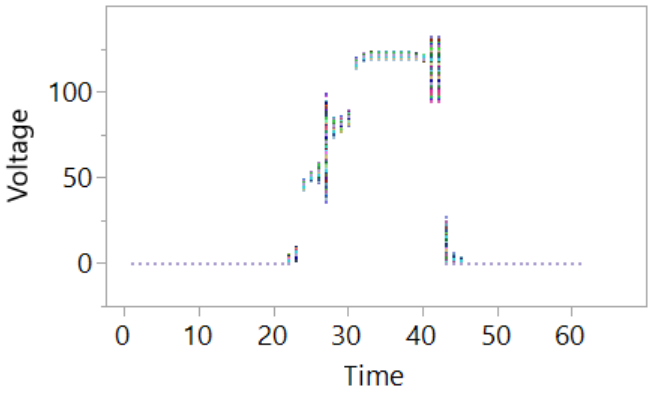


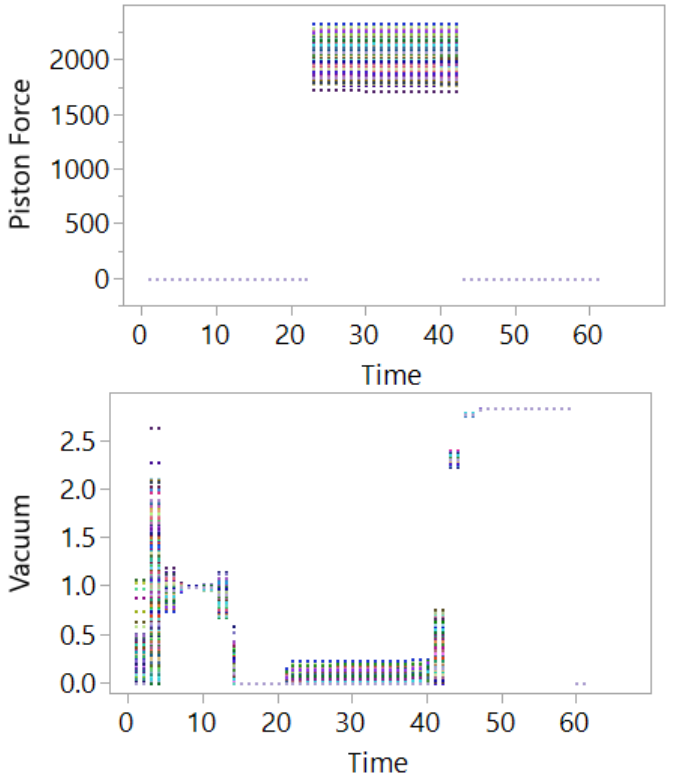
Scheme of anodic bonding procedure. The top tool works as a cathode and the chuck as anode.

- Picture courtesy of Wikipedia…

FIVE SENSOR STREAMS







INITIAL DATA PREPARATION

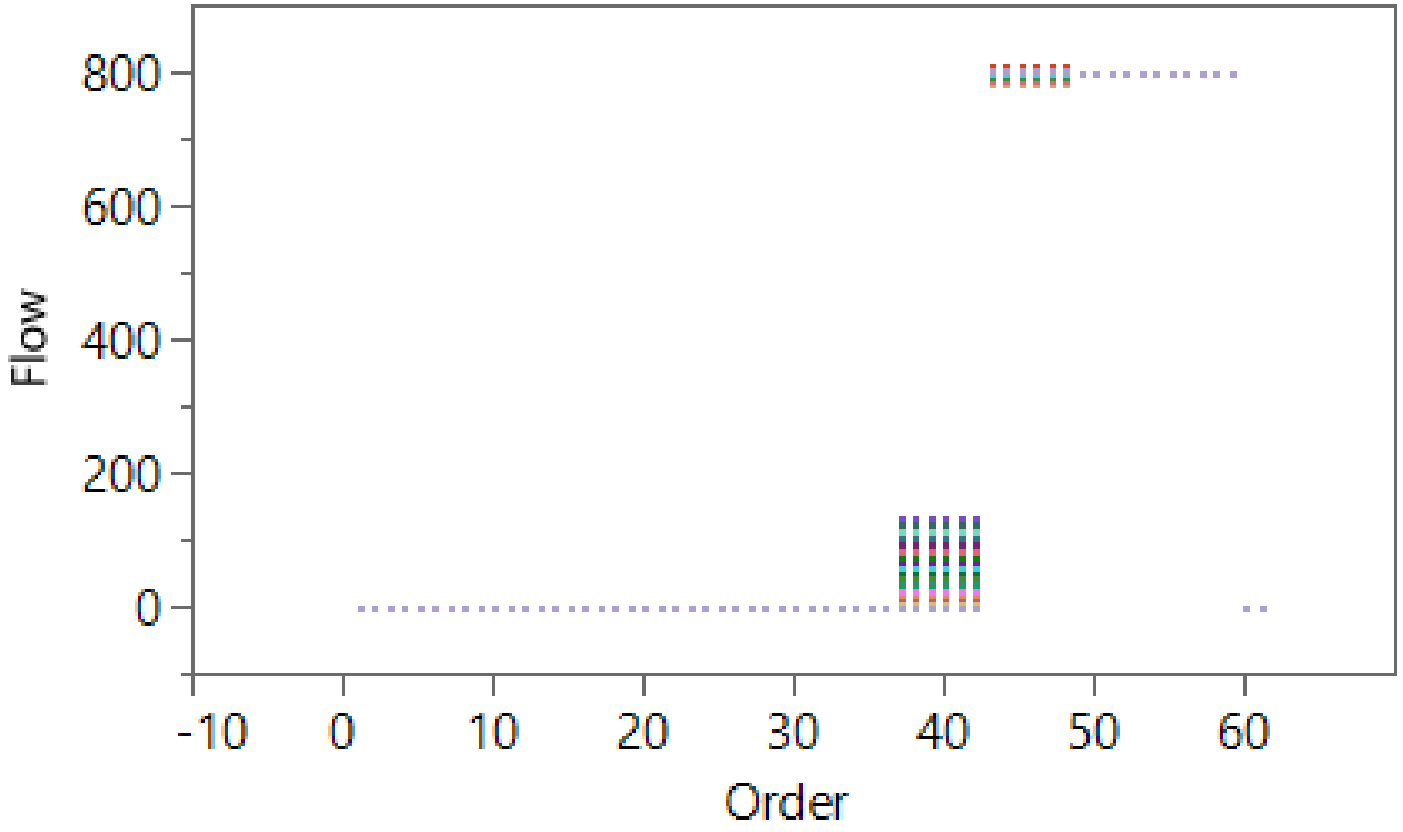
	Wafer Id	Condition	Validation	Time	Charge	Flow	Piston Force	Vacuum	Voltage
1	1	GOOD	Training	1	0.00	0.3013	0	0.00	0.00
2	1	GOOD	Training	2	0.00	0.3013	0	0.00	0.00
3	1	GOOD	Training	3	0.00	0.3013	0	0.50	0.00
4	1	GOOD	Training	4	0.00	0.3013	0	0.50	0.00
5	1	GOOD	Training	5	0.00	0.3013	0	0.94	0.00
6	1	GOOD	Training	6	0.00	0.3013	0	0.94	0.00
7	1	GOOD	Training	7	0.00	0.0008	0	0.99	0.00
8	1	GOOD	Training	8	0.00	0.0008	0	1.00	0.00
9	1	GOOD	Training	9	0.00	0.0008	0	1.00	0.00
10	1	GOOD	Training	10	0.00	0.0008	0	0.99	0.00
11	1	GOOD	Training	11	0.00	0.0008	0	0.99	0.00
12	1	GOOD	Training	12	0.00	0.0008	0	0.93	0.00
13	1	GOOD	Training	13	0.00	2.3e-6	0	0.93	0.00
14	1	GOOD	Training	14	0.00	2.3e-6	0	0.16	0.00
15	1	GOOD	Training	15	0.00	2.3e-6	0	0.00	0.00
16	1	GOOD	Training	16	0.00	2.3e-6	0	0.00	0.00
17	1	GOOD	Training	17	0.00	2.3e-6	0	0.00	0.00

- We sampled sensor data from 2000 wafers from the database
- Subsampled down to 61 equally spaced measurements
- Merged the sensor data with the result of the bonding step (Good/Bad)
- Used a “Grouped and Stratified” approach to create training & validation sets

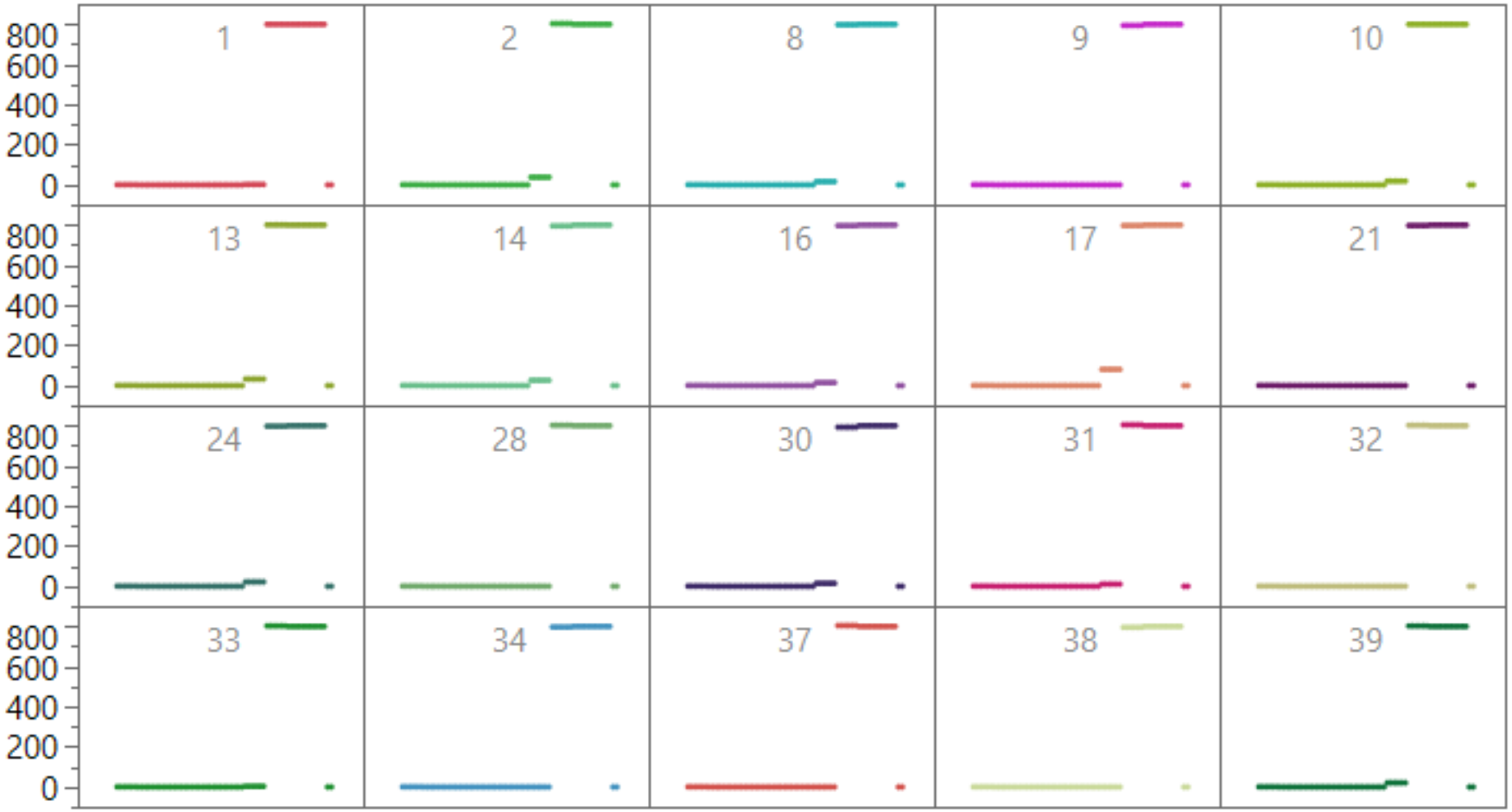
OVERALL APPROACH

- Perform data transformation/cleanup/alignment as needed
- Use functional principal components on each of the sensor streams as a dimension reduction scheme
- The combined FPC Scores will be used as inputs to models to predict Condition (Good/Bad)
- We choose the model that best predicts Condition on the validation set.
- Finally, we use the “Partition” trick to come up with a simple decision rule to classify new wafers

THE FLOW SENSOR STREAM



THE FLOW SENSOR STREAM



DATA CLEANUP

- The first step perform basic data cleanup:
  - Outliers
  - Error codes
  - Filter the streams down to time regions of interest

- Apply transformation and alignment operations as needed

- The Flow variable is in pretty good shape in this data set.

BASIS FUNCTIONS

- The next step is to find a basis function model to convert discrete measurements into continuous functions

- This makes it possible to combine functions with:
  - Irregular sampling times
  - Different sampling frequencies
  - Some amount of missing data

- This can be an ‘artful’ process that combines choices of
  - Basis function families
  - # Knots / Basis Functions
  - Knot Locations

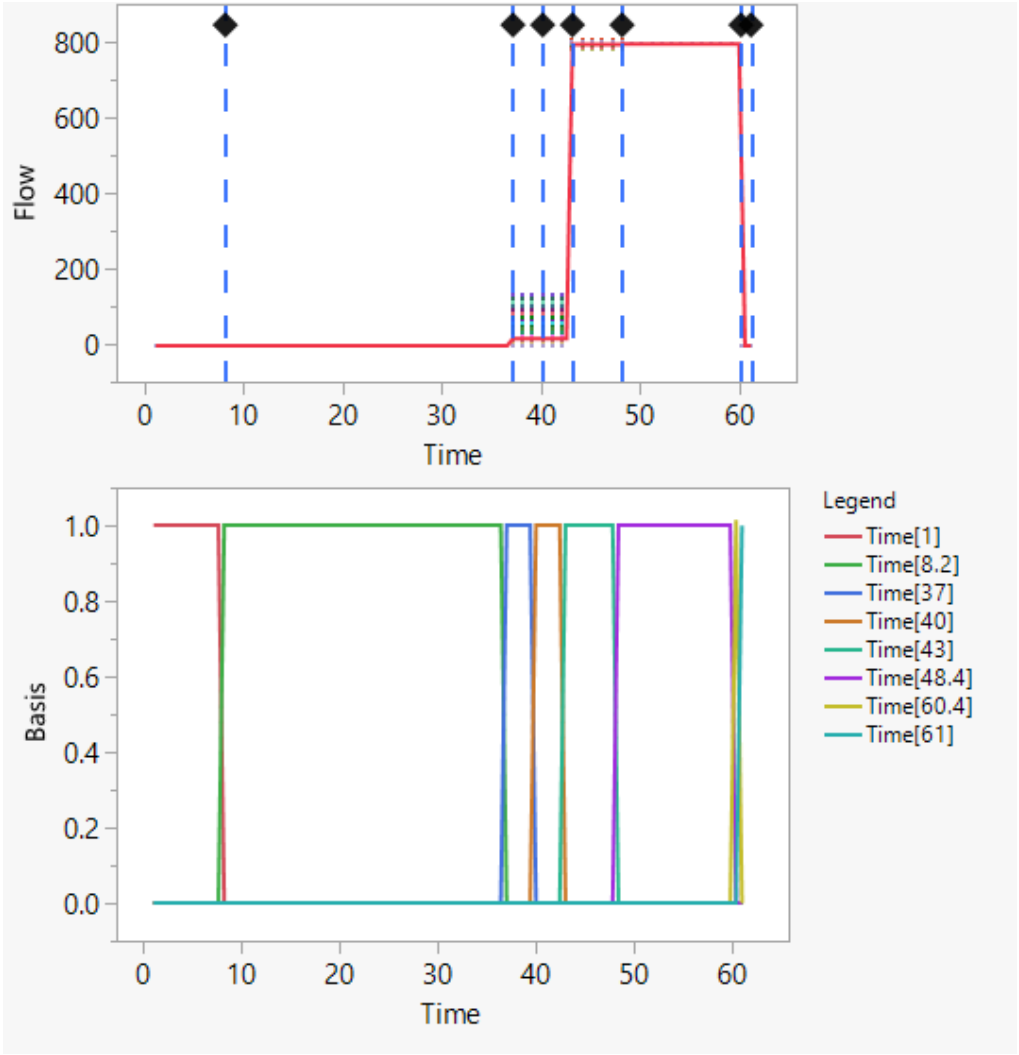
BASIS FUNCTIONS

- B-Splines
  - Useful in most F-DOE applications
  - “Simpler” functions
  - Situations where you want to customize the function carefully

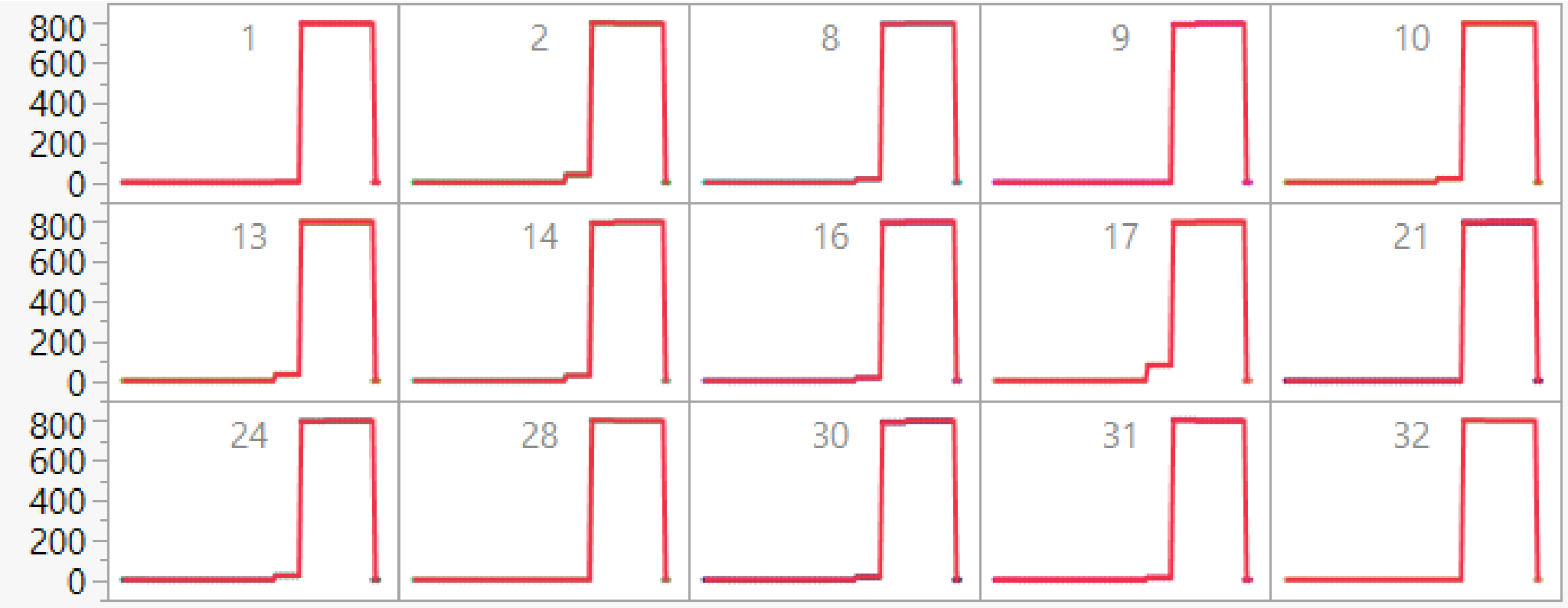
- P-Splines
  - Useful with more “complicated looking functions”
  - Often better as an automated procedure

- Fourier Basis
  - For heavily periodic data
  - These have been less useful than the splines in my limited experience…

7 KNOT CUSTOM B-SPLINE MODEL FIT



7 KNOT CUSTOM B-SPLINE MODEL FIT



7 KNOT CUSTOM B-SPLINE MODEL FIT

- This is really the result of fitting a variance components model to the data

- $\hat{f}(t_i) = \sum_j \hat{\beta}_j b_j(t_i) + \sum_j \hat{\gamma}_j b_j(t_i) + \varepsilon_i$
- $b_j(t)$ - the basis functions evaluated at the observed times
- β_j - basis function coefficients for the underlying mean function (fixed effects)
- $\gamma_j \sim Normal(0, \sigma^2_{\gamma_j})$ – function specific coefficients (BLUPs)
- $\varepsilon_i \sim Normal(0, \sigma^2_{\varepsilon})$ - residual variance

7 KNOT CUSTOM B-SPLINE MODEL FIT

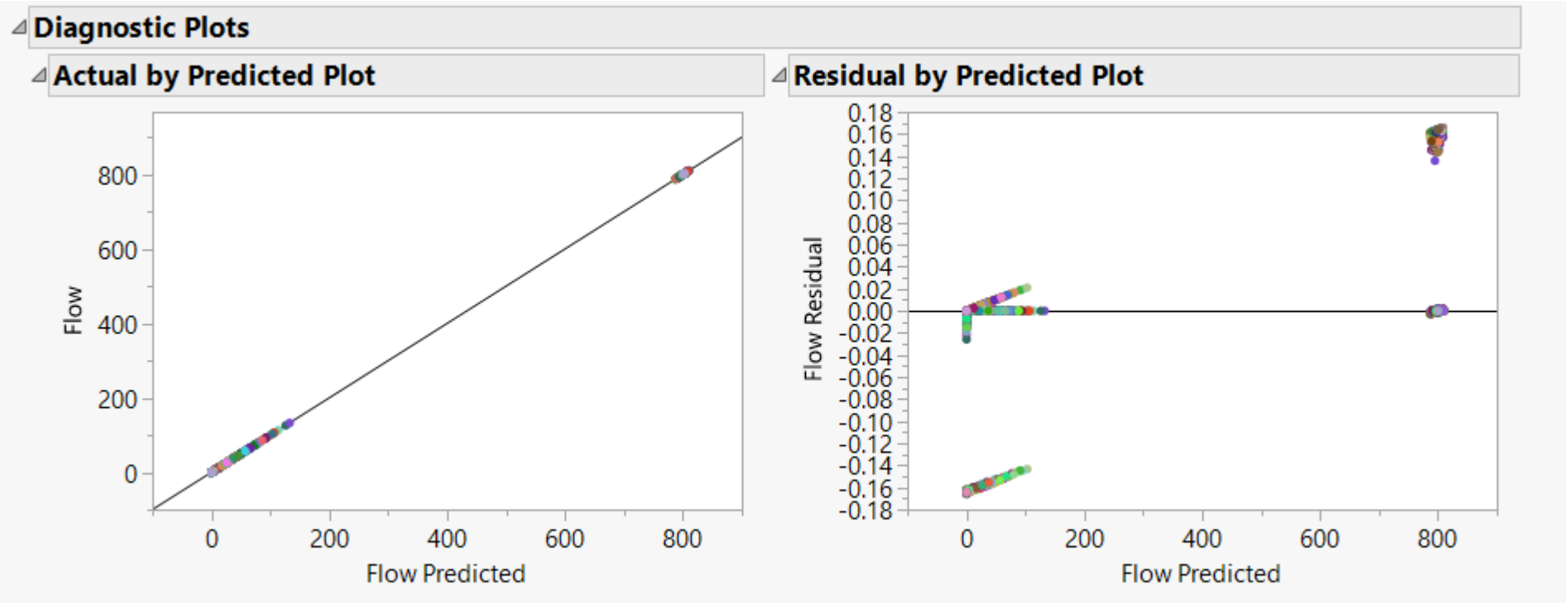
Basis Function Coefficients

Parameter	Mean Estimate	Standard Deviation Estimate
Time[1]	0.2583694	0
Time[8.2]	0.0067474	0
Time[37]	19.374697	19.324703
Time[40]	19.374697	19.324703
Time[43]	798.65285	3.9529603
Time[48.4]	799.97929	0.3501391
Time[60.4]	0	0
Time[61]	0	0

Random Coefficients by Function

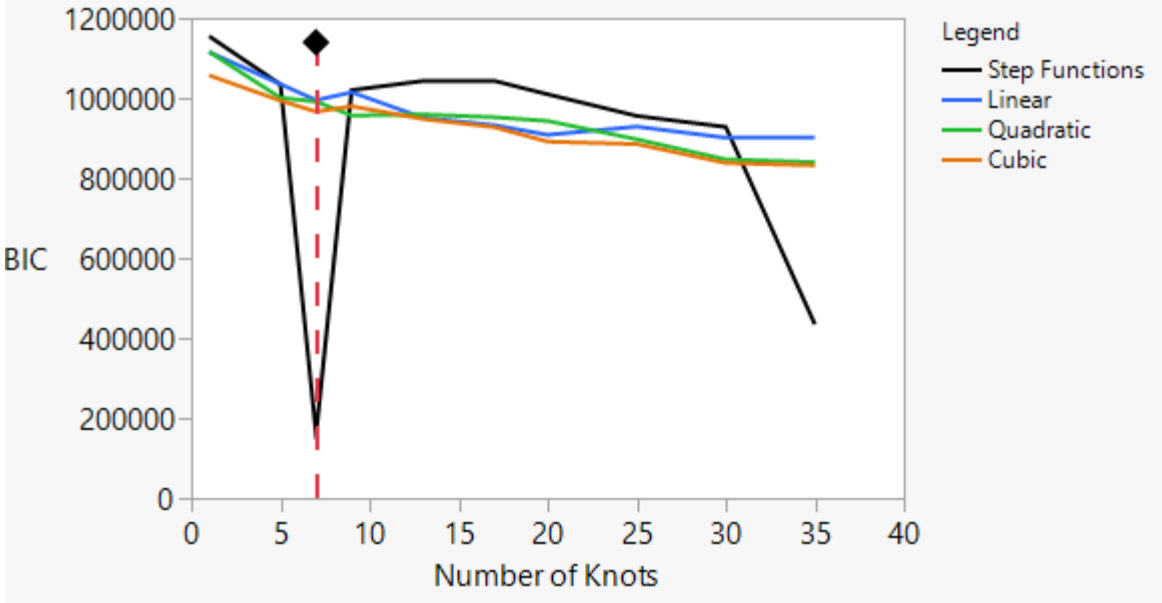
Wafer Id	Validation	Time[1]	Time[8.2]	Time[37]	Time[40]	Time[43]	Time[48.4]	Time[60.4]	Time[61]
1	Training	0	0	-17.54171	-17.54171	1.4722749	0.1237998	0	0
2	Training	0	0	18.634897	18.634897	4.8370952	0.3804477	0	0
8	Training	0	0	-2.962618	-2.962618	-0.54517	-0.046797	0	0
9	Training	0	0	-19.3698	-19.3698	-4.324241	-0.342897	0	0
10	Training	0	0	-0.205501	-0.205501	1.3933246	0.1093955	0	0
13	Training	0	0	13.028377	13.028377	1.9961551	0.152263	0	0
14	Training	0	0	7.1557475	7.1557475	-3.186925	-0.266051	0	0
16	Training	0	0	-4.8776	-4.8776	-1.624187	-0.133563	0	0
17	Training	0	0	61.230054	61.230054	-1.029891	-0.115761	0	0
21	Training	0	0	-19.3698	-19.3698	-1.549379	-0.117158	0	0
24	Training	0	0	1.9972952	1.9972952	-0.921634	-0.079664	0	0
28	Training	0	0	-19.3698	-19.3698	3.7194608	0.3473873	0	0

7 KNOT CUSTOM B-SPLINE MODEL FIT



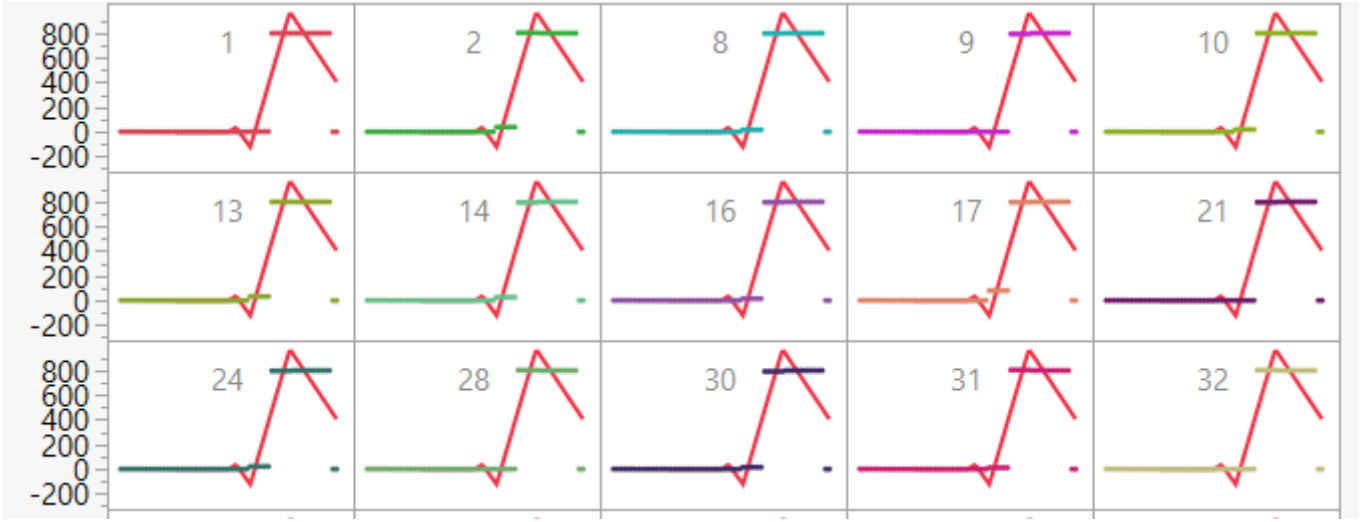
- Actual predicted plot is good – very tight around 45 degree line
- There are patterns in the residual by predicted plot, but residuals are very small…

OTHER B-SPLINE MODEL FITS



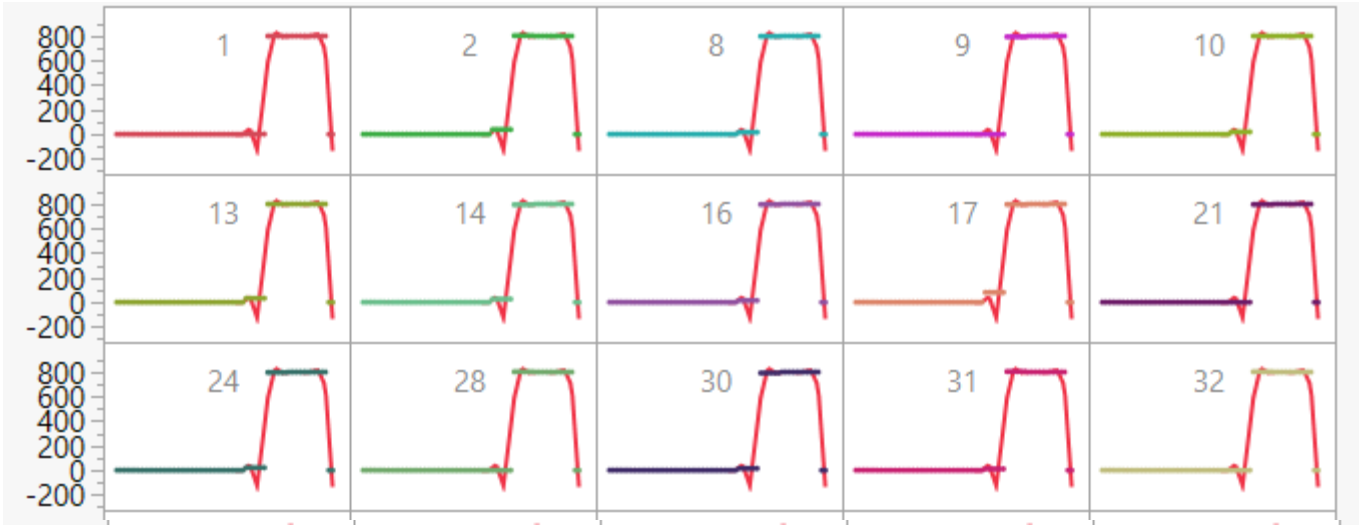
- We tried several other automatically generated knot locations, numbers, and orders
- The hand curated 7-Knot B-spline was the best by BIC…

OTHER B-SPLINE MODEL FITS



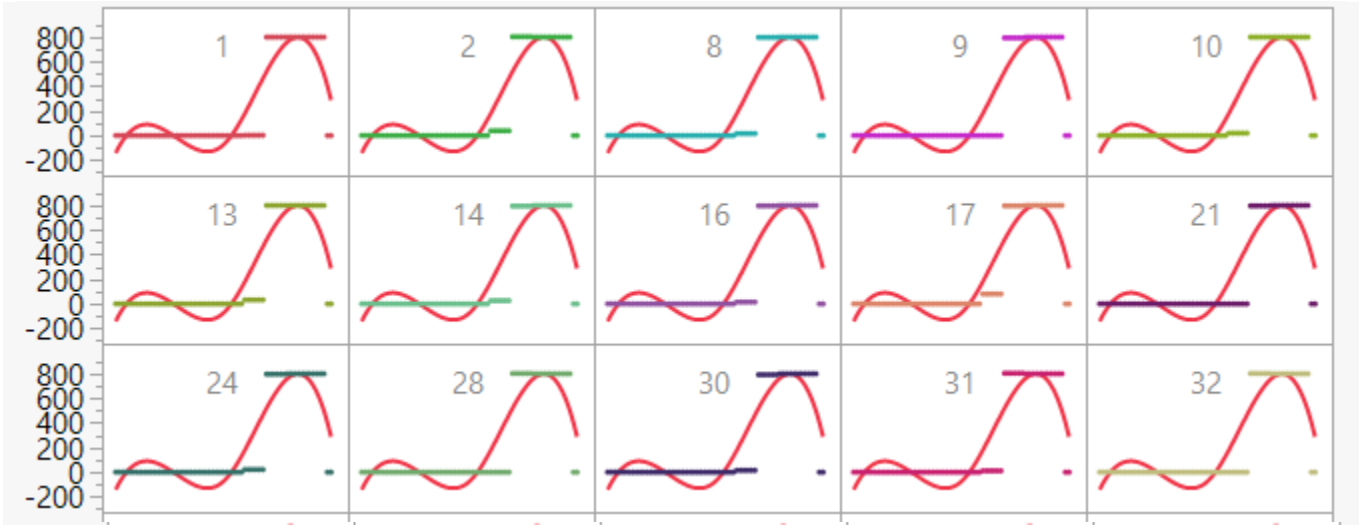
- Visual inspection usually makes it easy to rule out suboptimal spline models

OTHER B-SPLINE MODEL FITS



- Visual inspection usually makes it easy to rule out suboptimal spline models

OTHER B-SPLINE MODEL FITS



- Visual inspection usually makes it easy to rule out suboptimal spline models

FUNCTIONAL PRINCIPAL COMPONENTS

- Spline models to obtain a surrogate models is a crucial step in functional model, ***but not the end goal!***

- We need a reduction and feature extraction

- Functional Principal Components Analysis (FPCA) gives us a new basis expansion:

$$\hat{f}(t_i) \approx \hat{\mu}(t_i) + \sum_{j=1}^{D} \hat{S}_j \cdot \hat{v}_j(t_i)$$

$\hat{\mu}(t_i)$ - underlying *Mean Function* (across the different functions)

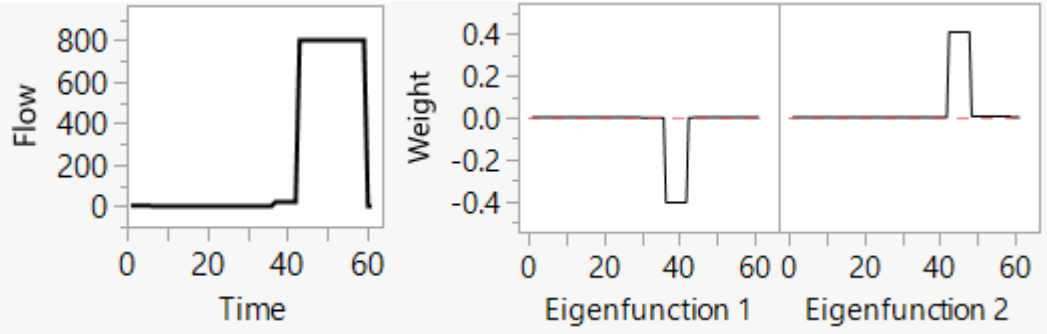
$\hat{v}_j(t_i)$ - orthogonal *Eigenfunctions*, chosen to maximize variation they explain

$\hat{S}_j$ - *FPC Scores*

FUNCTIONAL PRINCIPAL COMPONENTS

Functional PCA

FPC	Eigenvalue	20 40 60 80	Percent	Cumulative
1	3156.6		97.1%	97.1%
2	93.0		2.86%	100%



Wafer Id	FPC 1	FPC 2
1	43.14044	3.6012376
2	-45.82812	11.852175
8	7.2858672	-1.336398
9	47.635517	-10.59659
10	0.5055526	3.412344
13	-32.0404	4.8923018
14	-17.59854	-7.804334
16	11.995234	-3.979772
17	-150.583	-2.507047
21	47.63588	-3.79998
24	-4.912079	-2.257203
28	47.636735	9.1082651

- We reduced these measurements down to just 2 dimensions

- For example:
  - $\hat{f}_1(t) \approx \hat{\mu}(t) + 43.14 \cdot \hat{v}_1(t) + 3.601 \cdot \hat{v}_2(t)$
  - $\hat{f}_2(t) \approx \hat{\mu}(t) + -45.83 \cdot \hat{v}_1(t) + 11.85 \cdot \hat{v}_2(t)$

FUNCTIONAL PRINCIPAL COMPONENTS ALL SENSOR STREAMS

	Wafer Id	Condition	Validation	Flow FPC 1	Flow FPC 2	Charge FPC 1	Piston Force FPC 1	Vacuum FPC 1	Vacuum FPC 2	Vacuum FPC 3	Vacuum FPC 4	Voltage FPC 1	Voltage FPC 2	Voltage FPC 3	Voltage FPC 4
1	1	GOOD	Training	43.1412...	3.6012...	3075.413...	176.51598566	-0.18448912	0.0402560478	-0.026050017	0.0855265784	9.364924818	0.9803258523	3.482737...	-1.11020...
2	2	GOOD	Training	-45.828...	11.852...	1933.852...	909.73504643	-1.041899176	-0.358417486	-0.314611329	-0.247677719	-41.840283...	3.8272999116	-1.407366...	3.35382...
3	8	GOOD	Training	7.28597...	-1.336...	1509.270...	485.0765683	-0.966067254	-0.207984143	0.111556688	-0.204236432	-1.1016417...	-7.684377068	-4.099558...	0.86067...
4	9	GOOD	Training	47.6361...	-10.59...	-40.6718...	328.46361305	-0.567178691	0.1568307762	0.154856933	0.0276360574	-18.671139...	-3.288587053	-3.528775...	-1.88719...
5	10	GOOD	Training	0.50562...	3.4124...	1429.405...	-355.3000281	0.6861708189	-0.155363825	-0.072142318	0.0371732302	4.97857286...	-5.409718347	5.365950...	0.34087...
6	13	GOOD	Training	-32.040...	4.8924...	482.8843...	499.82000498	-0.970769525	-0.175257303	-0.029781684	-0.071832351	7.80070252...	-6.122699304	-0.280159...	-2.57304...
7	14	GOOD	Training	-17.598...	-7.804...	-1923.50...	-256.3366493	0.4003052655	0.0139557085	0.0355156802	-0.001896619	-10.984346...	5.1349285468	-7.171163...	-0.84730...
8	16	GOOD	Training	11.9953...	-3.979...	-3990.89...	-574.3577805	0.7672481506	0.0383742917	0.0406297589	-0.044793261	7.38151895	-0.962357021	-1.295879...	-1.56825...
9	17	GOOD	Training	-150.58...	-2.507...	1826.424...	147.28922423	-0.649458676	-0.480083187	0.1187989854	0.0817756817	-6.0658402...	-4.507527079	0.716658...	4.05312...
10	21	GOOD	Training	47.6366...	-3.800...	-2371.02...	-39.19812269	0.1193240776	0.0715183986	0.1201179238	-0.019081539	3.66817803...	-4.990741701	1.262970...	-0.40068...
11	24	GOOD	Training	-4.9122...	-2.257...	-659.697...	-169.4657207	0.2606149262	-0.102698442	0.1259056577	-0.021246042	6.58945128...	1.6856604787	-3.087383...	1.42431...
12	28	BAD	Training	47.6377...	9.1083...	-2337.92...	162.49853454	-0.287329739	0.4176634973	-0.049332944	0.0731561774	-30.828256...	0.2257734212	2.604980...	0.72117...
13	30	GOOD	Training	11.5479...	-14.42...	1083.430...	-31.99715798	-0.090059859	-0.092652519	0.1147380681	0.0206431458	-6.8425191...	-6.991691441	2.281762...	-0.40770...
14	31	GOOD	Training	21.6422...	14.228...	1216.352...	-480.5118377	0.8289567781	-0.070240107	0.1078100515	-0.076407204	3.01668389...	-3.235421257	0.93048169	3.32853...
15	32	BAD	Training	47.6376...	9.2615...	1104.619...	187.14750423	-0.338413284	0.1966873137	0.1039091751	0.0247536688	6.946648885	-2.873960265	0.380428...	-0.69192...
16	33	GOOD	Training	38.9990...	10.634...	539.1283...	-54.89198387	0.3197517142	0.1535076206	-0.065935006	0.0428017716	-13.252975...	-4.866510427	3.999512...	0.27373...

- We repeat this process for all responses and organize the FPCs together in one table
- There is one row per function, and small number of FPCs per sensor stream
- Use the merged FPCs as inputs to machine learning models

LASSO VS NEURAL NETWORK FPC MODELS

Binomial Lasso

Model Summary

Response	Condition
Distribution	Binomial
Estimation Method	Lasso
Validation Method	Validation Column
Probability Model Link	Logit

Measure	Training	Validation
Number of rows	1339	661
Sum of Frequencies	1339	661
-LogLikelihood	278.45511	155.28576
Number of Parameters	36	36
BIC	816.09863	544.34665
AICc	630.95629	386.84075
Generalized RSquare	0.5135505	0.4376448

Model NTanH(8)NTanH2(8)

Training

Condition

Measures	Value
Generalized RSquare	0.5041603
Entropy RSquare	0.4155417
RMSE	0.2540286
Mean Abs Dev	0.1333366
Misclassification Rate	0.0851382
-LogLikelihood	282.8382
Sum Freq	1339

Validation

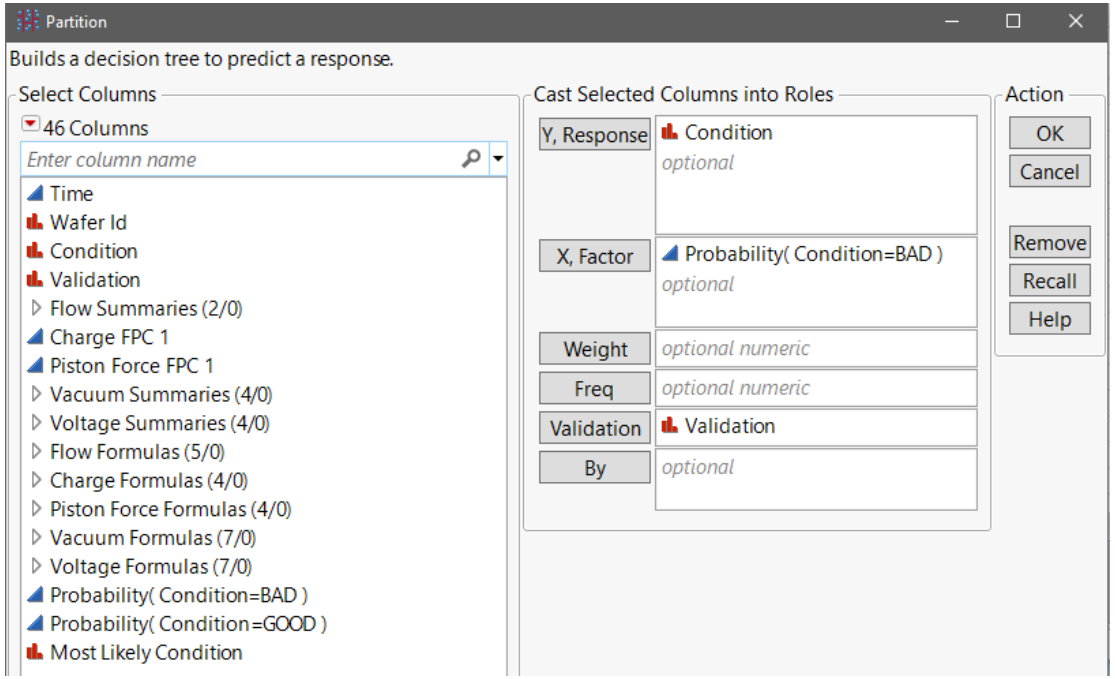
Condition

Measures	Value
Generalized RSquare	0.4824964
Entropy RSquare	0.3945205
RMSE	0.2558232
Mean Abs Dev	0.1340082
Misclassification Rate	0.086233
-LogLikelihood	145.25143
Sum Freq	661

- The two layer NN model performed somewhat better on the validation set than the Binomial Lasso model (with interactions)

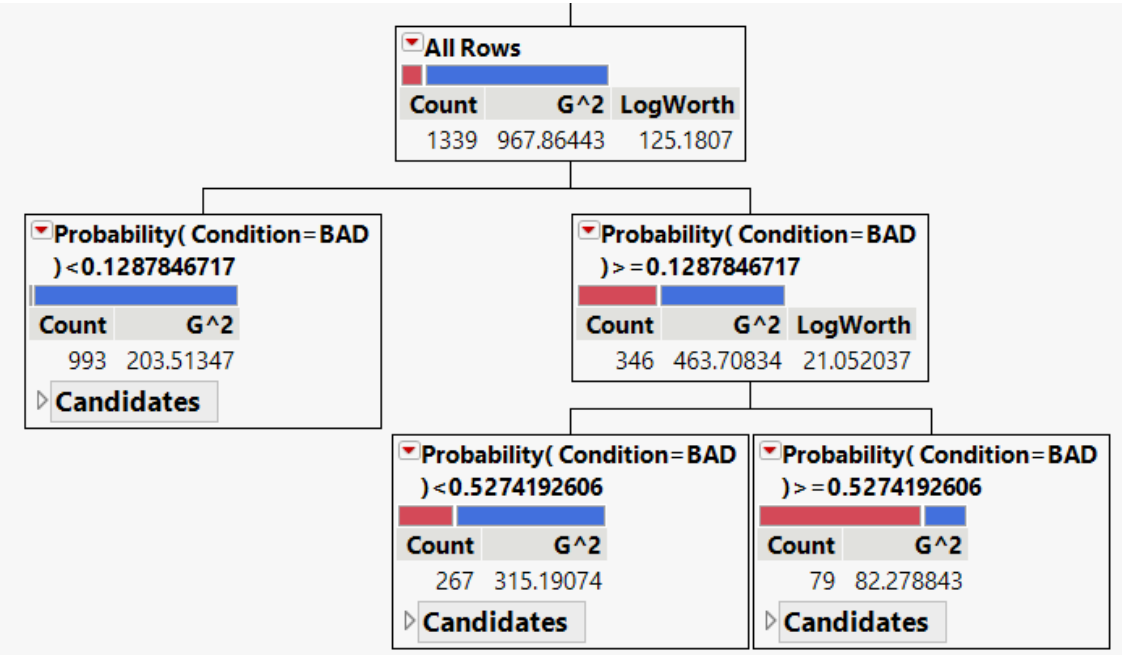
- We are primarily interested in prediction, so we move forward with the NN model

PROBABILITY BASED PREDICTIONS OF RARE(ish) EVENT MODELS



- We can get this with a regression tree model using Condition as the response, the same training/validation sets and the neural network probability as the input

TURNING PROBABILITIES INTO DECISIONS



- This leads to an easily operationalizable red light / yellow light / green light system

- If Pr(Bad) < .12 it is almost certainly good,
- If Pr(Bad) >.53 it is probably bad.
- If .12 <= Pr(Bad) <=.53 you should inspect the wafer for damage.

CHARACTERIZING BAD WAFERS

Parameter Estimates for Original Predictors

Term	Estimate	Std Error	Wald ChiSquare	Prob > ChiSquare	Lower 95%	Upper 95%
Intercept	3.8659381	0.20001	373.59958	<.0001*	3.4739257	4.2579505
Vacuum FPC 2	-10.71303	0.9339098	131.58761	<.0001*	-12.54346	-8.882603
Charge FPC 1	0.0003816	7.7212e-5	24.425899	<.0001*	0.0002303	0.0005329
Charge FPC 1*Vacuum FPC 2	-0.001205	0.0003691	10.655983	0.0011*	-0.001928	-0.000481
Voltage FPC 2	0.0803526	0.0318658	6.3584429	0.0117*	0.0178968	0.1428083
Vacuum FPC 3*Voltage FPC 1	0.1854952	0.0758514	5.9805078	0.0145*	0.0368292	0.3341612
Flow FPC 1*Vacuum FPC 4	0.0504178	0.0245667	4.2118682	0.0401*	0.002268	0.0985675
Vacuum FPC 4*Voltage FPC 3	-0.680375	0.3415157	3.9689443	0.0463*	-1.349733	-0.011016
Charge FPC 1*Voltage FPC 1	8.6366e-6	4.5353e-6	3.6263837	0.0569	-2.524e-7	1.7526e-5
Flow FPC 1*Flow FPC 2	-0.000951	0.0005247	3.2821269	0.0700	-0.001979	0.0000778

- Although we didn't use the Lasso model, it does give clues into what is driving bad wafers – it looks fairly clear that Vacuum and Charge are predictive

CHARACTERIZING BAD WAFERS