Variable Selection and Sensitivity Analysis via Dynamic Trees with an application to Computer Code Performance Tuning

Robert B. Gramacy
University of Chicago Booth School of Business

faculty.chicagobooth.edu/robert.gramacy with
M.A. Taddy and S. Wild

SIAM UQ Meeting
Raleigh, NC — April 5 2012
Partition Trees

Recursive, axis-aligned, splits divvy up input space into independent predictive models.

Prediction surface is dictated by tree structure and leaf model.
Use of trees in regression dates back to AID (Automatic Interaction Detection) by Morgan and Sonquist in 1963.

A breakthrough with CART (Breiman, et al., 1984): algorithms for fitting partition trees for classification and regression.

The key idea: Tree structure allows us to obtain flexible response surfaces with simple predictive models at the leaves.

- Classification leaves: optimal class choice.
- Regression leaves: constant or linear mean response.

The hard part is fitting partition structure (depth, splits, etc).
Bayesian Tree Regression Models

Chipman, George, & McCullogh (1998): Bayesian tree models
Penalize tree depth with a prior on each split node:

\[ p_{\text{split}}(\eta, T) = \alpha (1 + D_\eta)^{-\beta} \]

Then all that’s left to specify is the leaf model.

- **Constant (TC) leafs:** \( E[Y|\eta(x)] = \mu_{\eta(x)}. \)
  
  *(Chipman, George, and McCulloch, 1998)*

- **Linear (TL) leafs:** \( E[Y|x, \eta(x)] = \mu_{\eta(x)} + x' \beta_{\eta(x)}. \)
  
  *(Chipman, George, and McCulloch, 2002)*

- ...
Average predictions over both tree and parameter uncertainty.

Fit the model in an MCMC scheme, with repeated sampling:

- Randomly choose part of tree to alter (prune, grow, etc).
- Accept the move via Metropolis-Hastings.

Sales pitch... Trees are:

Fast nonstationary heteroskedastic regression at little computational/modeling expense.

The price paid is discontinuity in your response surface.
Where are we now?

There are some problems with the status quo.

Tree-space is complicated.

We move through small changes. It is very difficult to explore possible partition structures for large high-dimensional datasets.

MCMC is time consuming.

Many nonparametric regression and classification applications beg for sequential inference!
Dynamic Regression Trees

But suppose that we are able to come up with a dynamic characterization of partition trees ...

Use sequential Monte Carlo:

- Approximate the posterior with a finite set of tree particles.
- Propagate uncertainty through a resample/reweight step for each new observation.

The posterior only changes a little bit with every new observation, and tree-space complexity builds in time.
\[ \eta(x_{t+1}) \]

\[ x_1 < 5 \]

\[ x_2 < 4 \]

\[ T_t \]

\[ x_{t+1} \]

\[ \eta(x_{t+1}) \]

\[ x_2 \]

\[ x_1 \]

\[ T_t \]

Stay Prune Grow

Stay Prune Grow
Inference by Sequential Monte Carlo

Resampling involves the predictive probability of the next observation in the appropriate leaf of each particle.

Propagating each particle proceeds as follows:

- Build the three possible trees: $\mathcal{T}^{\text{stay}}, \mathcal{T}^{\text{prune}}, \mathcal{T}^{\text{grow}}$
- Draw the new tree $\propto \pi(\mathcal{T}^{\text{move}}) p(y^t \mid x^t, \mathcal{T}^{\text{move}})$

Only calculate for the relevant subtree (neighborhood of $\eta(x_t)$)

Division of labor: resampling incorporates global changes to the tree posterior, while propagation provides local modifications.
Dynamic TC and TL

30 passes in random order
Larger Comparison

Friedman’s data with 200 training and prediction on 1000.

\[ \mathbb{E}\{Y|x\} = \\
10 \sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 \\
x_6 \ldots x_{10} \text{ irrelevant} \]
**Optimization by EI**

**Simple optimization:** Predict over a space-filling grid and choose to execute the location that maximizes EI.
Classification

We’ve done linear and constant leaves; the other great tree application is with multinomial leaves for classification:

\[ p(y = c) = p_c(x), \quad c = 1, \ldots, C. \]

Model summary counts for each leaf node as

\[ z_\eta \sim MN(p_\eta, |\eta|) \]

Classification is easy!
Example: UCI credit approval

- 690 credit card applicants grouped by approval (‘+’ or ‘-’)
- 15 input variables (11 categorical, binary encoding expands the input space to 47 dimensions)

We did 100 independent repetitions of 10-fold CV.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Dynamic Tree</th>
<th>TGP soft-max</th>
<th>GP soft-max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Miss-class</td>
<td>0.136 (0.038)</td>
<td>0.142 (0.036)</td>
<td>0.146 (0.04)</td>
</tr>
<tr>
<td>CPU hours per fold</td>
<td>0.01</td>
<td>1.62</td>
<td>5.52</td>
</tr>
</tbody>
</table>
Dynamic Variable Selection

Trees naturally “select” variables/interactions for data mining.

- Tree prior is a natural regularizer.
- Inputs not split upon are not useful.

But,

- Particles may contain spurious splits on useless variables.
- Counting the number of splits is not sufficient for measuring relative importance.
We can measure the importance of predictors through their contribution to a reduction in the predictive uncertainty at each split location $\eta$:

$$J_k(T) = \sum_{\eta \in I_T} \Delta(\eta) \mathbb{1}_{[v(\eta) = k]}, \quad k = 1, \ldots, p$$

where

- $\Delta(\eta) = \int_{A_\eta} \sigma^2_\eta(x) \, dx - \int_{A_{\eta \ell}} \sigma^2_{\eta \ell}(x) \, dx - \int_{A_{\eta r}} \sigma^2_{\eta r}(x) \, dx$ for regression

- $\Delta(\eta) = |A_\eta| H_e - |A_{\ell}| H_\ell - |A_r| H_r$, where $H_\eta = - \sum_c \hat{p}_c \log \hat{p}_c$ for classification
Posterior relevance statistics for the Friedman Data:

Variable selection can then be based on the probability that the relevance is positive: \( \mathbb{P}(J_k(T) > 0) \approx \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\{J_k(T_t^{(i)}) > 0\} \)
A classification problem ...
Automatic tuning of executables

New fads in cloud computing require automatic optimization source codes. Choosing:

- compiler and flags
- source translations, unrolling loops and making other rearrangements

can lead to dramatically faster executables on some architectures.

But getting it wrong can lead to slow ones, and the search spaces can be huge.

Variable selection and sensitivity analysis can narrow the search, providing rules of thumb.
Consider a linear algebra kernel experiment performed at Argonne (Balaprakash et al., 2011).

**Six inputs:**

- **two** loop unrolling parameters in \{1, \cdots, 30\}
- **three** compiler options (binary) for scalar replacement, loop parallelization and vectorization.
- **one** *cold-cache* effect parameter from repeated runs.

The code was run on a dense design of $2^3 \times 30^2$ design points, **199** resulted in a compile or runtime error.

The successful inputs were re-run **34** more times to examine the cold-cache effect.
A more realistic “compile time” experiment:

- space-filling design of size 500
- two repeats

What really matters?
Having reduced the search space by 98%, we perform a final EI-like optimization.
Round-up for trees

Dynamic trees can provide superior performance and are less expensive than alternatives.

▶ Models which can be fit sequentially.
▶ Lead to robust and flexible prediction.
▶ Ideal for sequential design and optimization.
▶ Useful for variable selection and sensitivity analysis.
▶ Adaptable to online inference.

That is, if you can live with a little discontinuity.

dynaTree package on CRAN