

# Sequential Optimization of a Computer Model: What's so special about Gaussian Processes anyway?

Hugh Chipman, Acadia University

*joint work with Vivi Wang and Pritam Ranjan (Acadia)*

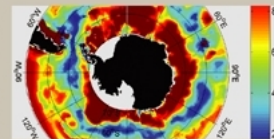
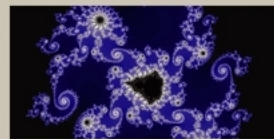
*Statistical Society of Canada*

*May 26, 2010*



*Mathematics and Statistics at Acadia*

FACULTY OF PURE AND APPLIED SCIENCES



## Outline:

1. Motivating example: tidal power
2. Possible surrogates for the computer model
3. BART as a surrogate for nonstationary process
4. Performance in tidal power application
  - 1D turbine placement
  - 2D turbine placement
5. Discussion for computer experiments

## Motivating problem: tidal power generation

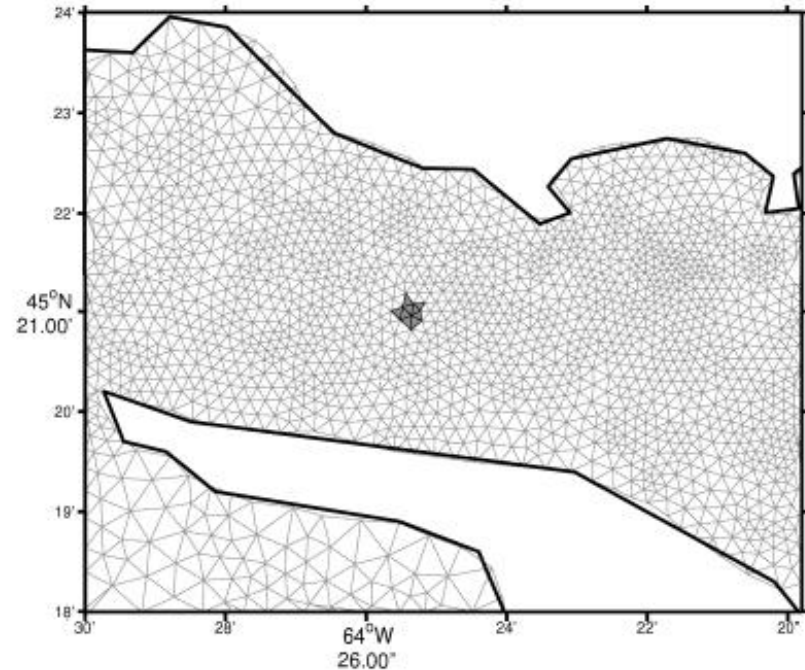
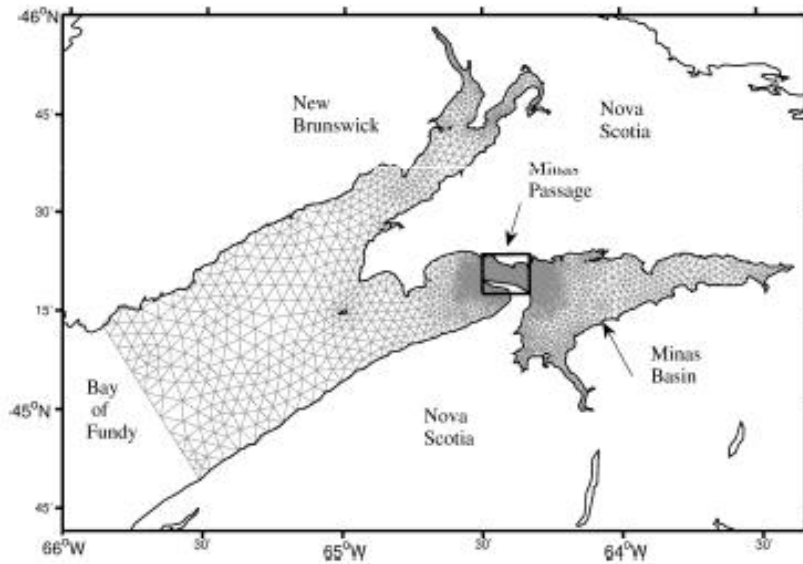


Bay of Fundy: highest tides in the world.  
(come visit us for SSC 2011)

## Motivating problem: tidal power generation



## Motivating problem: tidal power generation



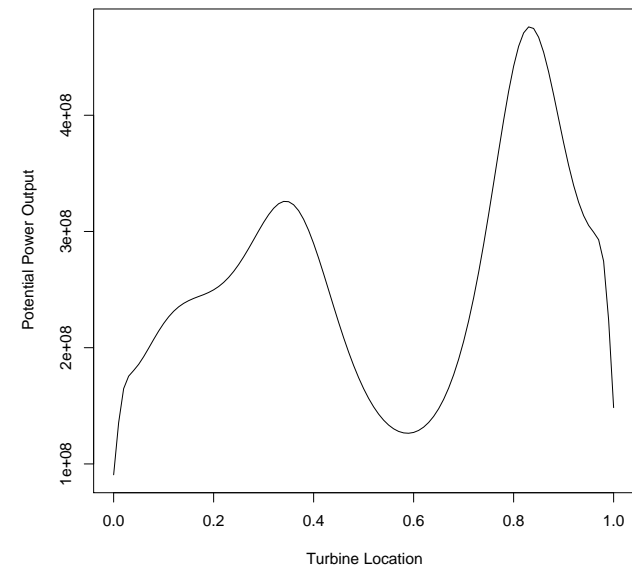
**The problem:** Place turbine to get maximum power.

**Where we come in:**

- Fluid dynamics models can predict flow (and power).
- Complex computer codes are time consuming.
- Adding turbines to the system changes the flow (requiring additional runs).

## Simplifications (for today)

- Simplified flow model (runs faster)
- Placement of a “fence”: a collection of turbines arranged in a line across the flow direction.
- But: we might consider 2D location of single turbines.
- Resultant “power function” has 1-dimensional input (position along flow axis)
- Plot at right is power generated for placement of a single fence of turbines.
- Plot evaluated by brute force computation.



## Sequential fence placement

Suppose we want to place 10 fences. A possible (brute force) algorithm is:

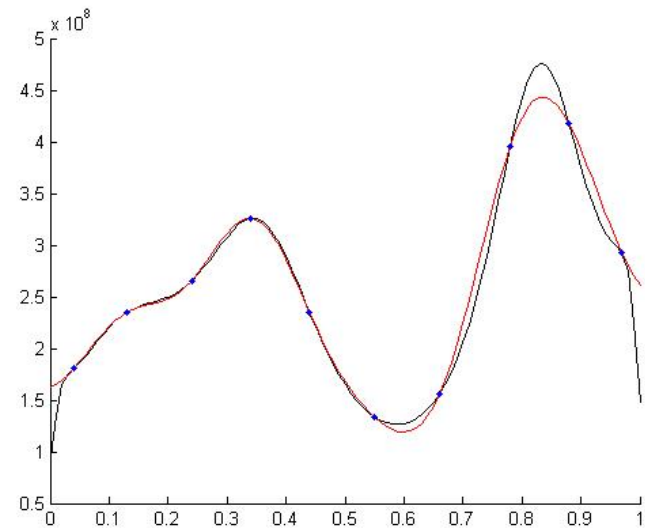
1. for  $i$  in 1:10 fences
2.     for  $j$  in 1:100 possible fence locations.
3.         Calculate total power with fixed locations for fences  $1, \dots, i - 1$  and fence  $i$  at location  $j$ .
4.     Pick best location for fence  $i$



# Approximating the power function

Brute force approach can be expensive.

- Every time we add a fence, the power function changes.
- Instead of an exhaustive grid evaluation, approximate the power function with a statistical model.
- Gaussian process (GP) models a natural choice.
- GP approximation with 10 evaluations at right.



## Sequential design for the surrogate

Estimate the black-box power function with a few function evaluations.

Sequential design approach (Jones, Schonlau & Welch 1998):

- To find a global maximum, sample the next design point to maximize expected improvement
- Improvement =  $I(x) = \max(y(x) - f_{\max}, 0)$
- Here  $f_{\max}$  is the current best estimate of the maximum,  $y(x)$  is the random response at  $x$ .
- Expected improvement:

$$E[I(x)] = s(x)\phi(u) + (\hat{y}(x) - f_{\max})\Phi(u)$$

where  $u = \frac{\hat{y}(x) - f_{\max}}{s(x)}$ ,  $\hat{y}(x) =$  prediction,  $s(x) =$  standard error of prediction

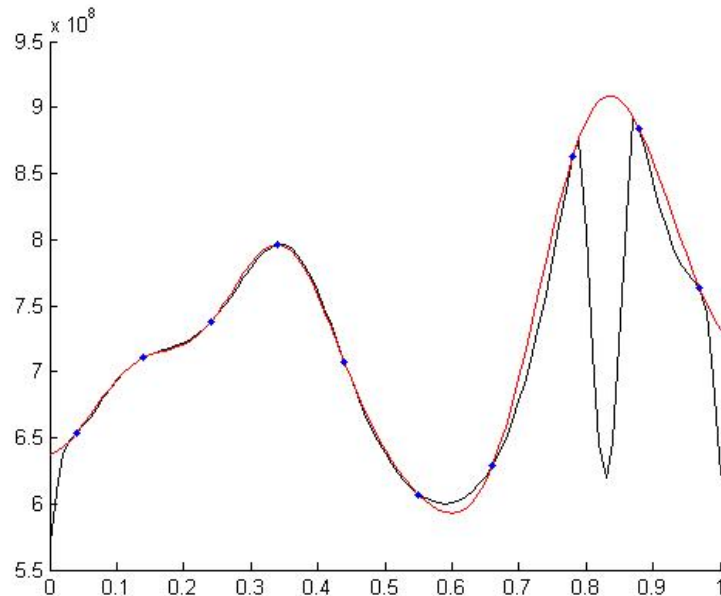
- Expected improvement = global criterion + local criterion
- EI will add points where uncertainty is large and/or we might have a optimum.

## Problems in paradise



This should work well.  
But it doesn't always.

(If you come to SSC 2011, you can visit Paradise. It's just 80 km from Acadia.)

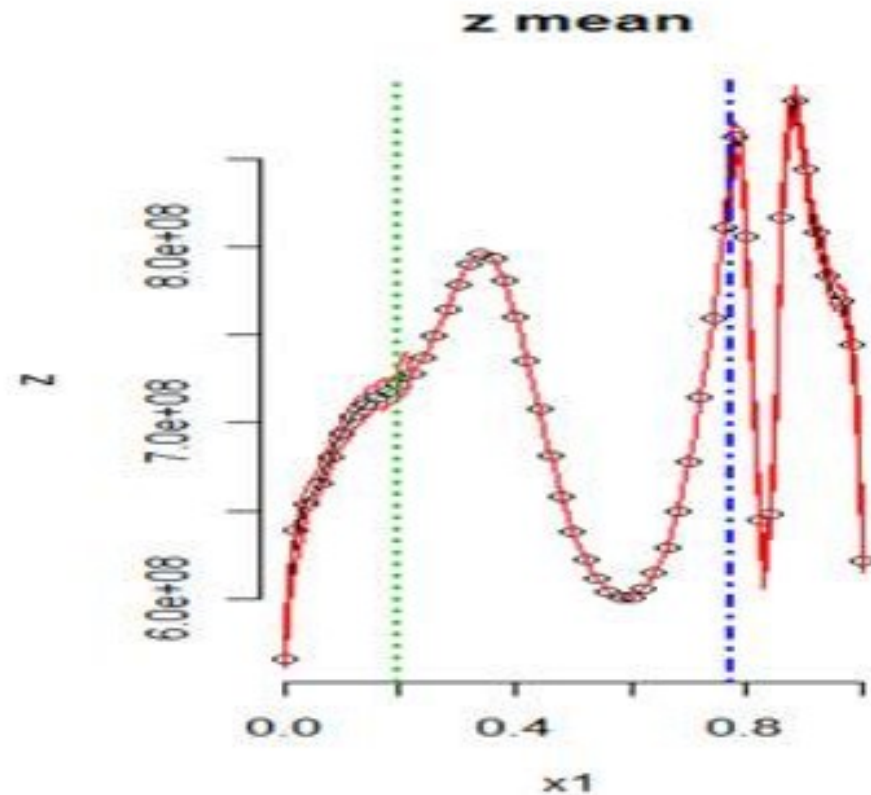


For the placement of turbine 2, the power function is nonstationary. GP models have difficulty with this.

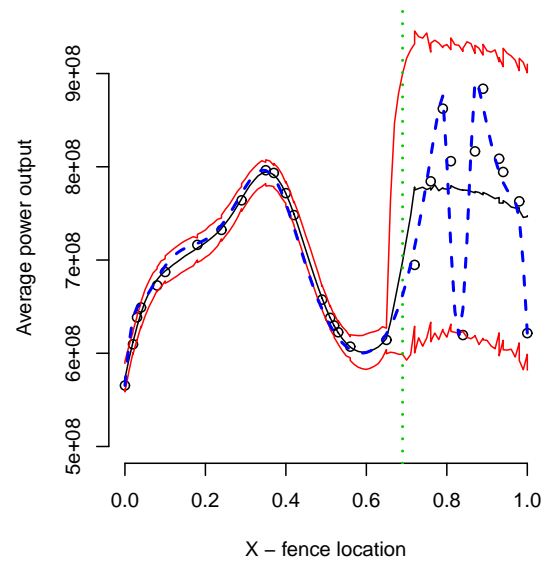
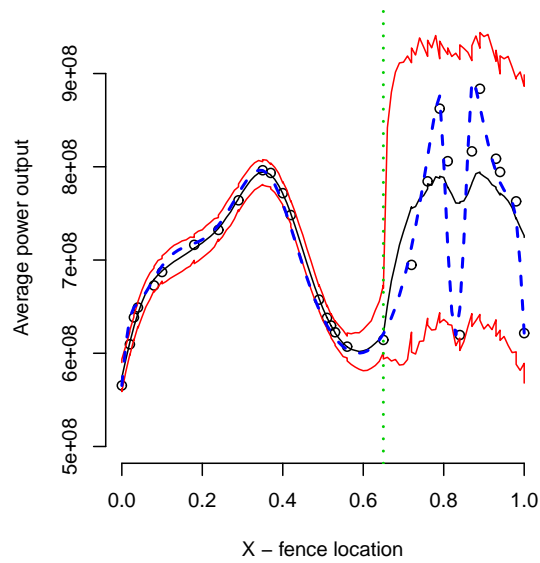
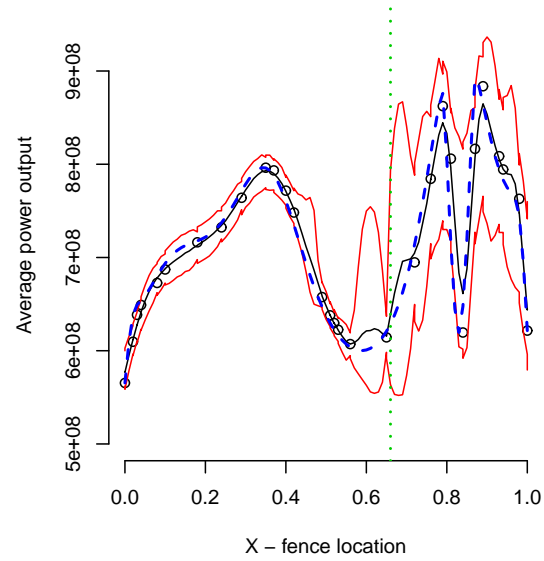
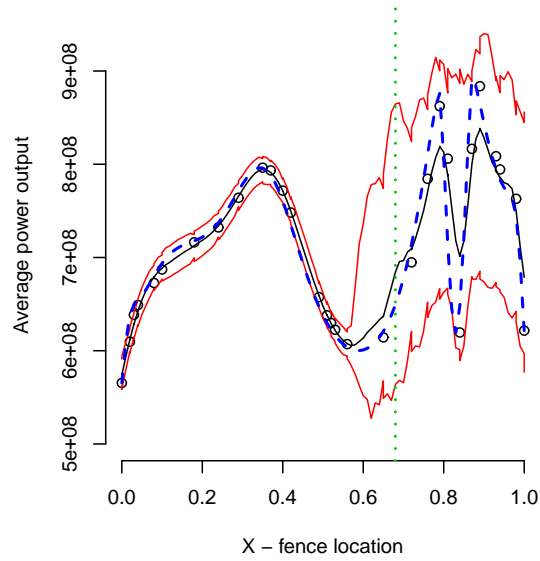
# Alternate approaches

Treed Gaussian Processes (Gramacy & Lee 2008)

- Decision tree partitions the  $x$  space into rectangles.
- Separate GP model in each rectangle.



# But TGP can fail...



## Abandoning GPs for a less stationary model

We really just need  $\hat{y}(x), s(x)$  for EI.

Additional requirements:

- Flexible regression model.
- Ability to interpolate observed  $y$  if necessary.

Bayesian Additive Regression Trees (BART, Chipman George and McCulloch 2007, 2010) meets these requirements.

Our data model is

$$\hat{y}(x) = g(x; T_1, M_1) + g(x; T_2, M_2) + \dots + g(x; T_m, M_m) + \sigma Z,$$
$$Z \sim N(0, 1).$$

Here  $T_i$  is a tree,  $M_i$  is a collection of constants associated with terminal nodes.

# BART

$$\hat{y}(x) = g(x; T_1, M_1) + g(x; T_2, M_2) + \dots + g(x; T_m, M_m) + \sigma Z,$$

- Bayesian formulation and MCMC estimation yield  $\hat{y}(x), s(x)$ .
- Each  $g \approx$  a function of a small subset of the  $x$ 's.
- Basically a sum of piecewise constant functions.
  - We lose on continuity.
  - We gain on nonstationarity (jumps possible).
- Note: Perhaps overkill for the univariate problem, but we'll see it works.

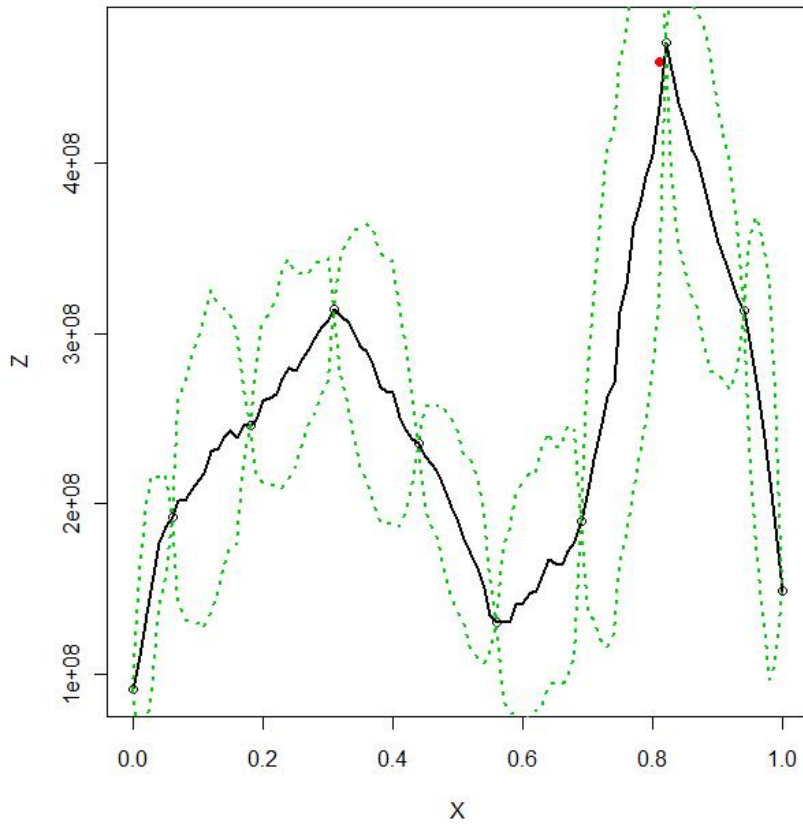
# BART

- Many details we won't cover.
- Priors represent uncertainty over parameters.
  - Effectively represent uncertainty over a space of functions.

Prior specification important for computer experiments:

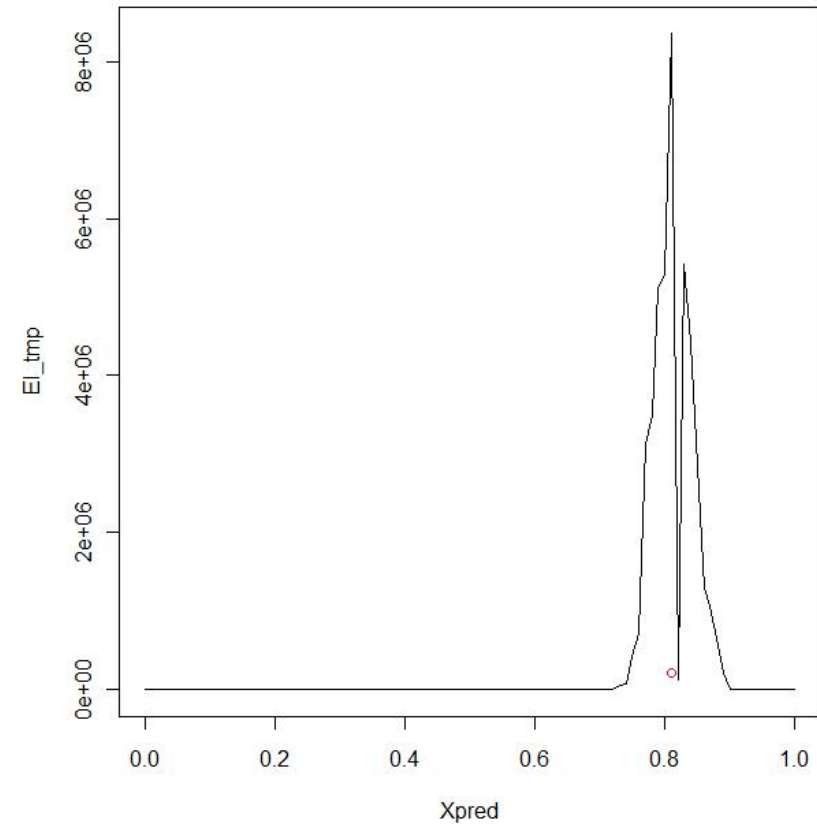
- $\hat{y}(x)$  should be allowed to interpolate observed  $y$ :
  - Less shrinkage of predictions toward a central mean.
- Residual variance  $\sigma \approx 0$ :
  - Set prior for  $\sigma$  to have high probability near  $\sigma = 0$ .

# Example of BART prediction and corresponding EI



$$\hat{y}(x) \pm s(x)$$

This is for placement of first turbine.



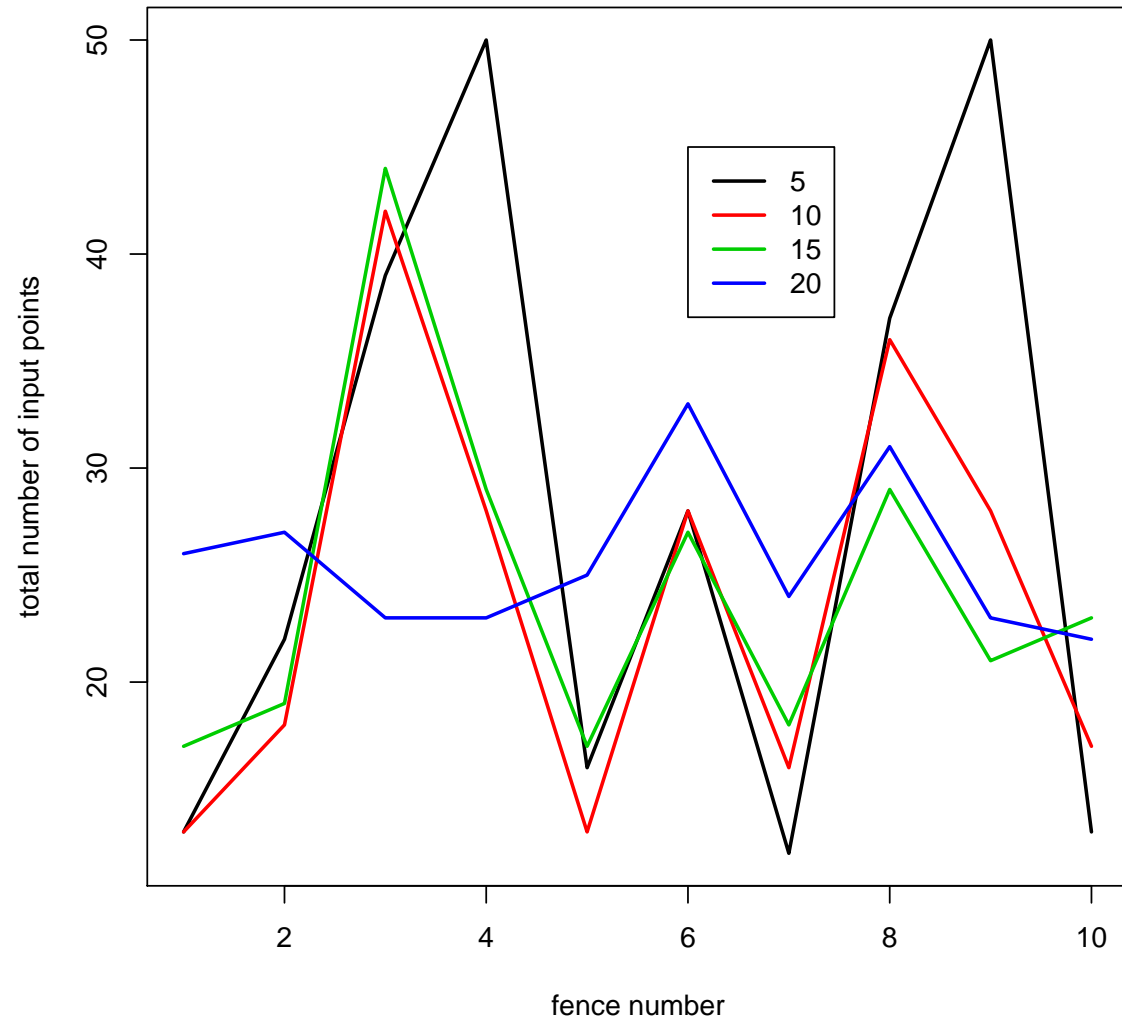
Expected Improvement

**A movie ...**

# Results

Placement of turbines 1-10, total # function evaluations for various size starting designs.

20 starting points has less variability.



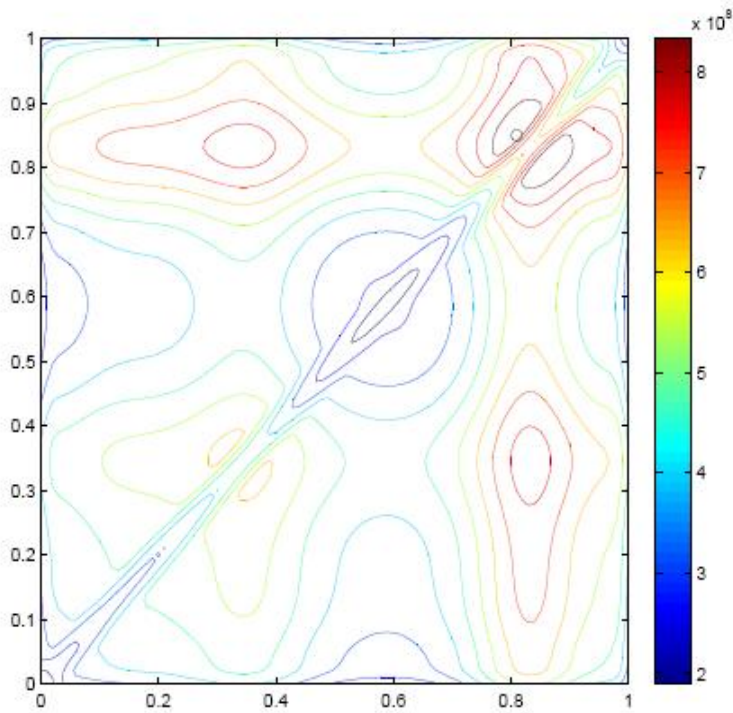
## 2D problem

- Place two turbine fences at a time.
- Conditional on first two fences, place next pair, and so on....
- Computer experiment is now predicting a power surface as a function of two locations.
- Statistical model for bivariate function (in triangular region due to symmetry).

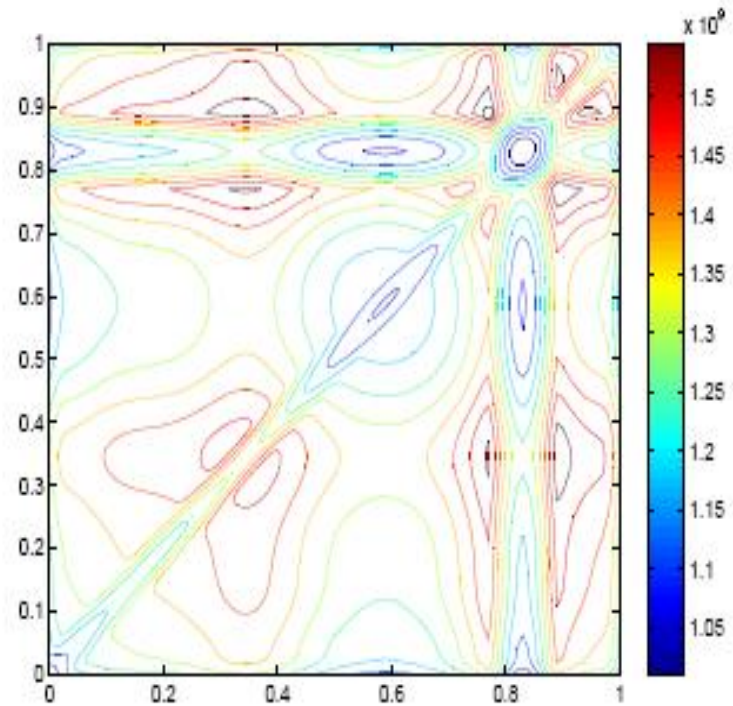


## 2D problem

Statistical model for bivariate function (in triangular region due to symmetry).



First pair

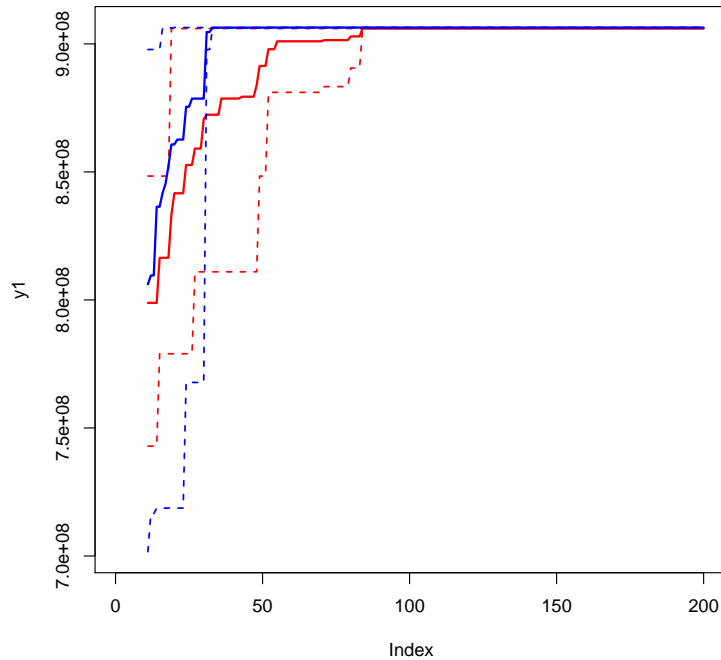


Second pair

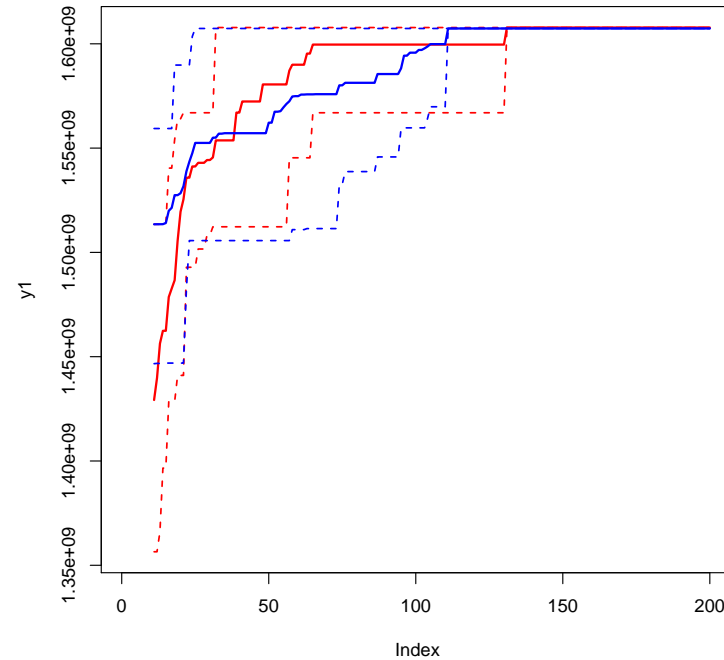
## 2D problem - Results

Predicted maximum vs. # function evaluations.

turbines 1+2



turbines 3+4



Not surprisingly the 2D problem requires more function evaluations to locate the optimum.

- Considerably more than twice as many evaluations as in 1D case.

## Practical conclusions

Power generated by 10 turbine fences, placement by 1D or 2D approaches.

Fence	1 at a time		2 at a time	
	1D Location ( $x_1$ )	Total Power ( $10^8\text{W}$ )	2D Location ( $x_1, x_2$ )	Total Power ( $10^8\text{W}$ )
1	0.83	4.76		
2	0.87	8.92	(0.82, 0.86)	9.06
3	0.79	12.82		
4	0.91	16.03	(0.78, 0.90)	16.08
5	0.34	19.03		
6	0.38	21.74	(0.33, 0.37)	21.86
7	0.29	24.40		
8	0.75	26.96	(0.29, 0.94)	27.00
9	0.96	29.38		
10	0.24	31.57	(0.24, 0.41)	31.46

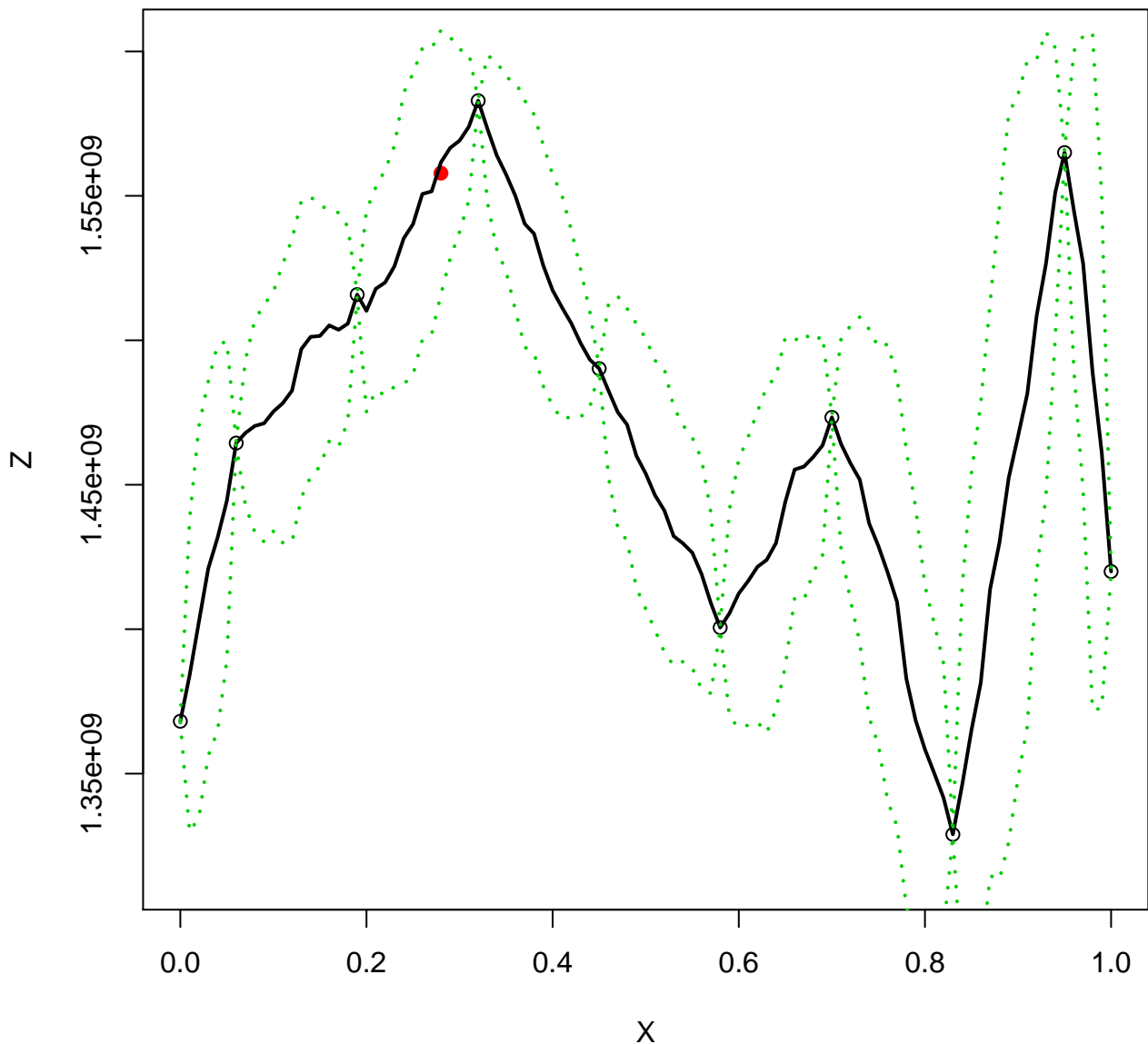
2D generally better, though not much difference.

## Discussion

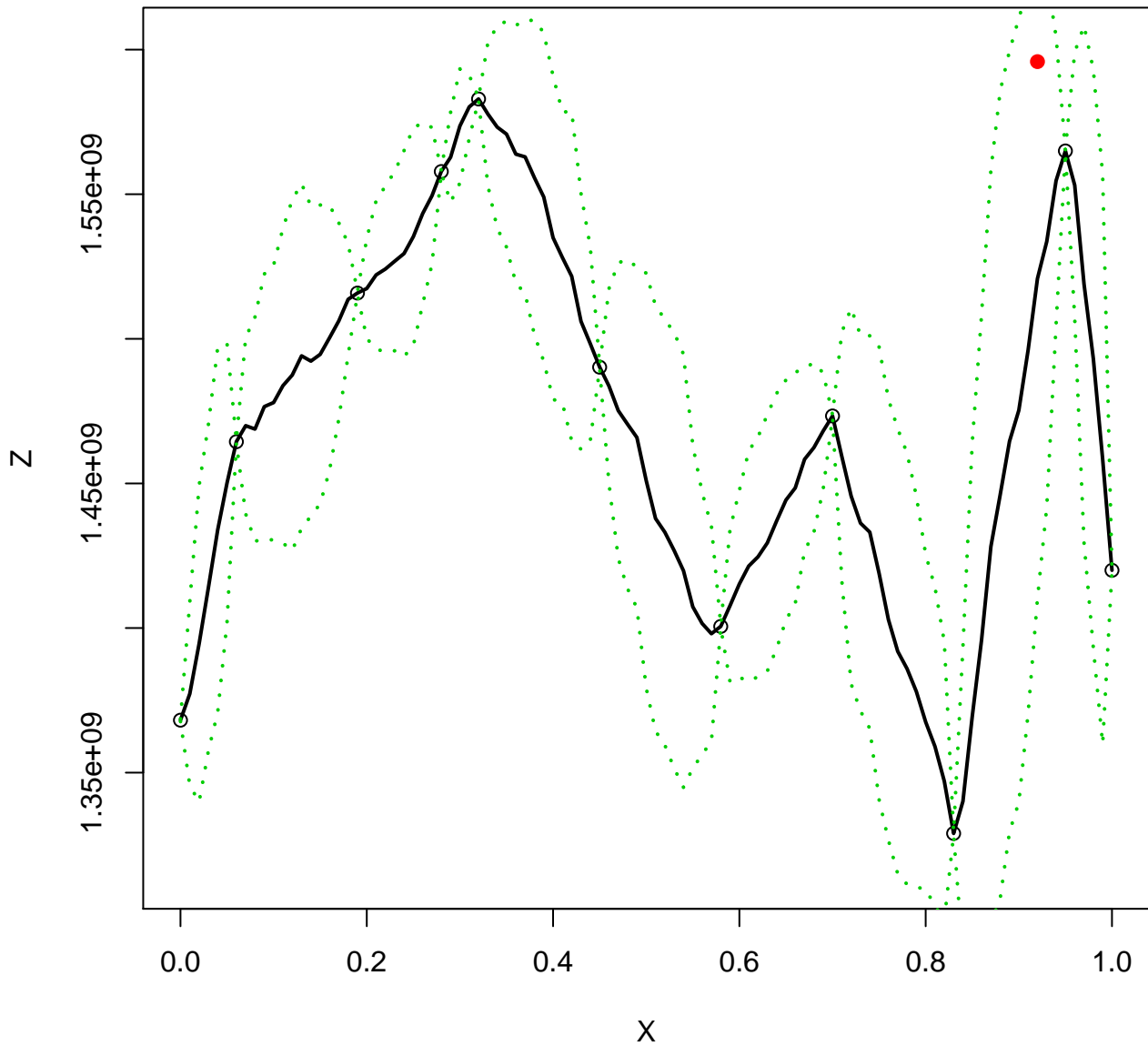
- We're cheating with EI (but it seems to still work).
  - EI criterion assumes  $\hat{y}(x)$  is normal. It isn't for us.
  - Alternatives include using MCMC output to calculate expectation.
- Small study with simple model - grow to larger problems
- When might BART be useful?
  - Nonstationary processes
  - Many X's (mixed types OK)
  - Many observations
  - Batch-sequential design possible: BART gives joint predictive distribution of  $\hat{y}(x)$ .
- Nothing special about BART either, any flexible model plausible.

Movie: BART estimation of power function for placement of 4th Turbine

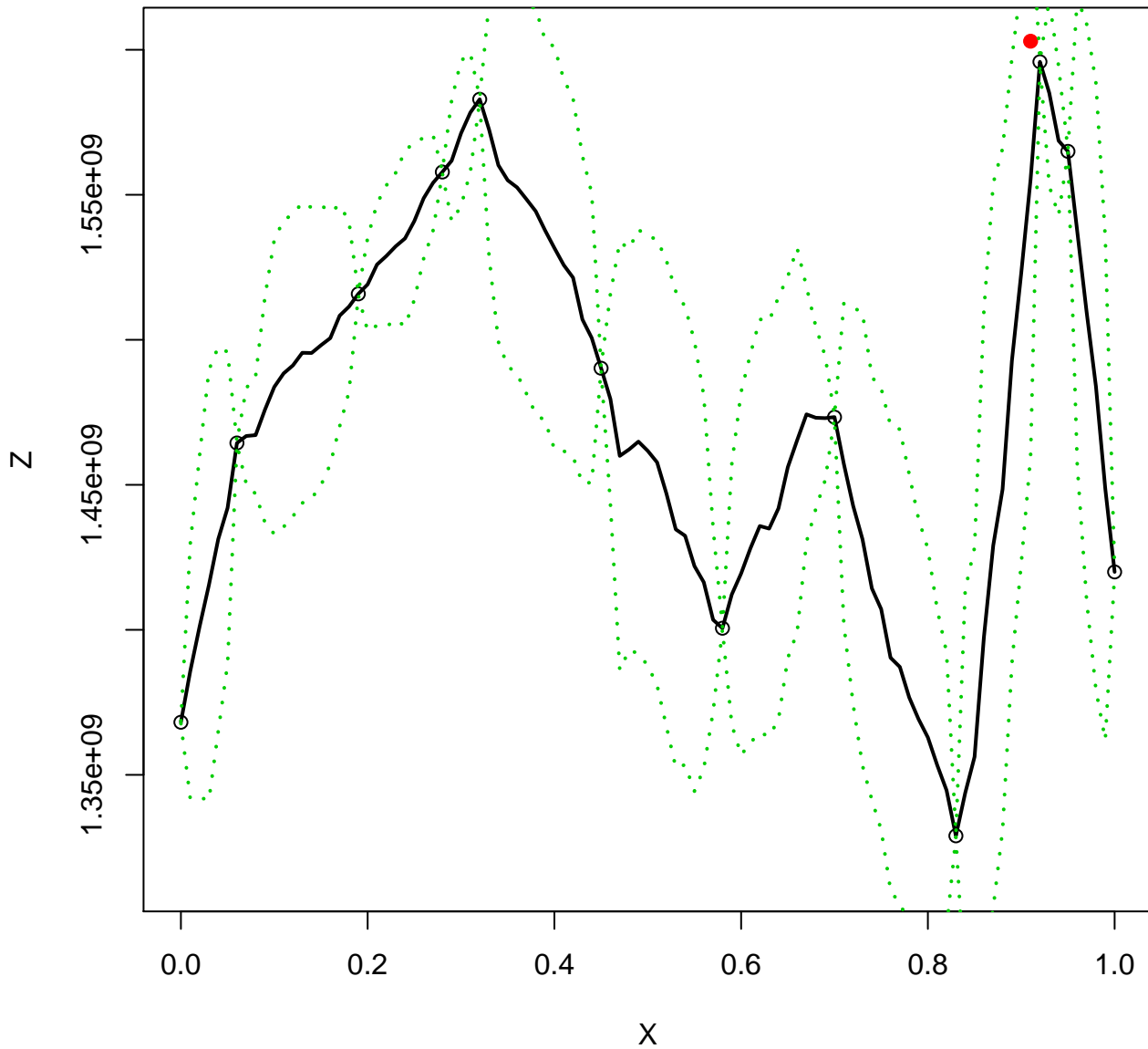
**BART fit (n0 = 10, k = 1)**



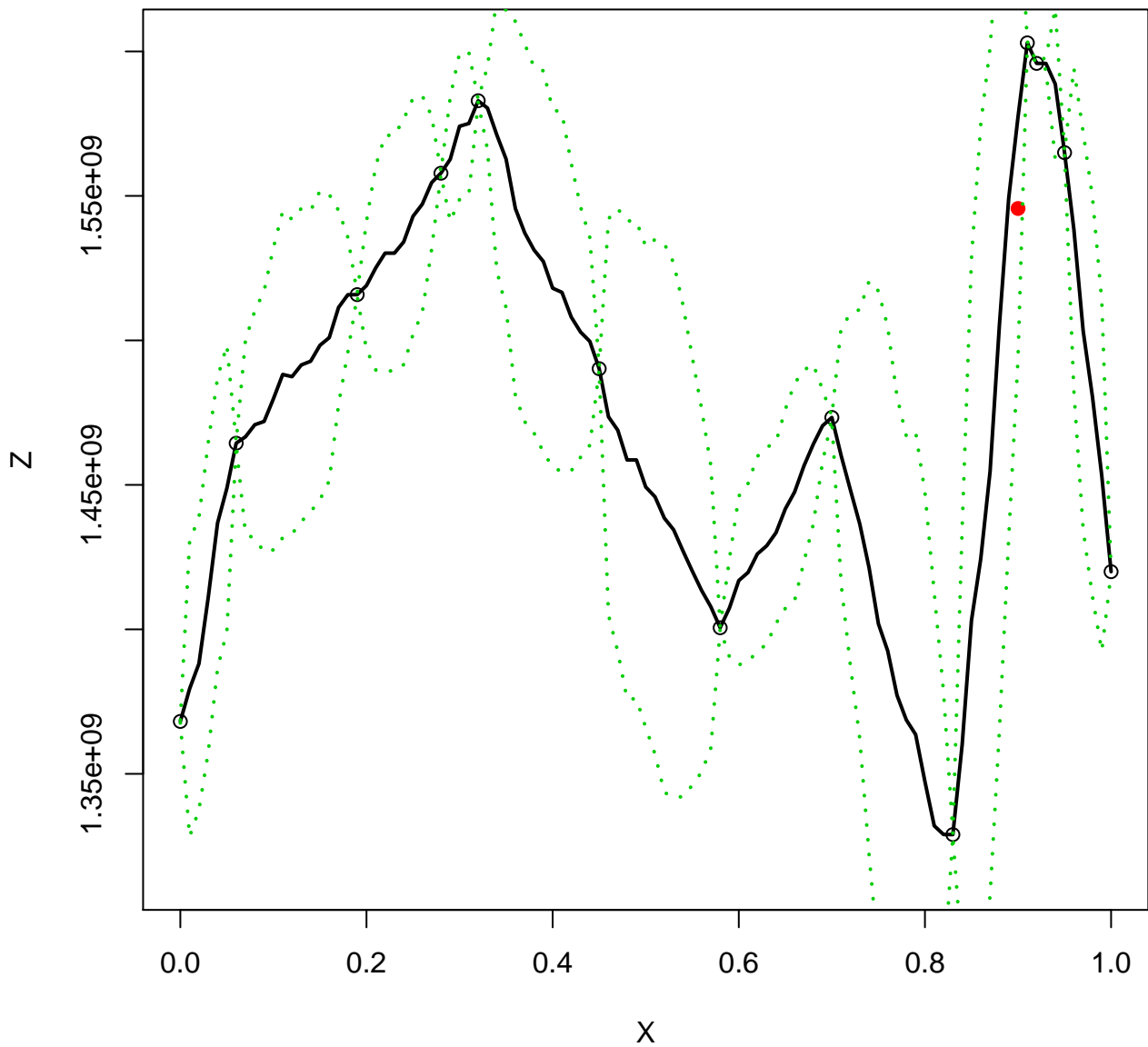
**BART fit (n0 = 10, k = 2)**



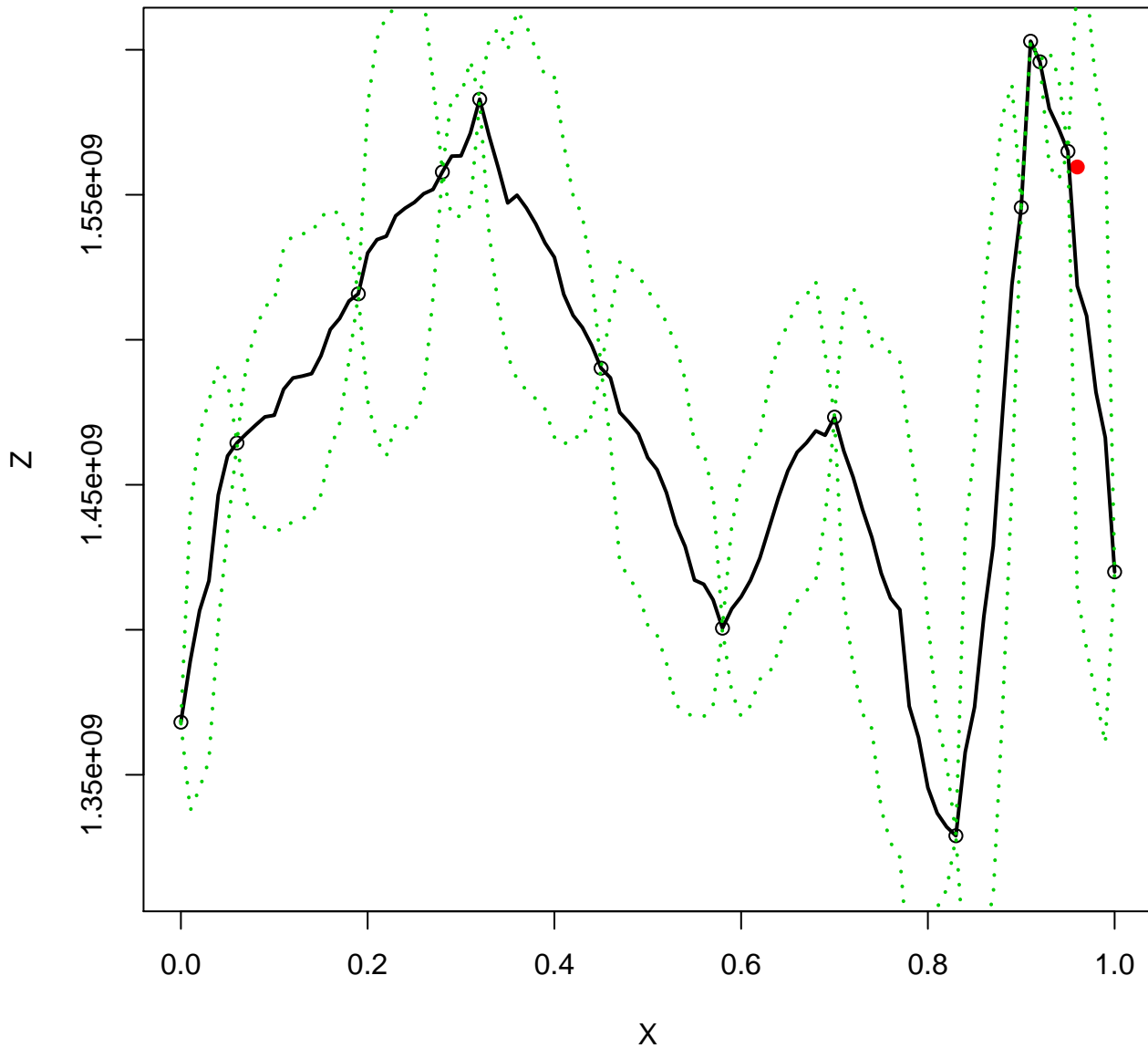
**BART fit (n0 = 10, k = 3)**



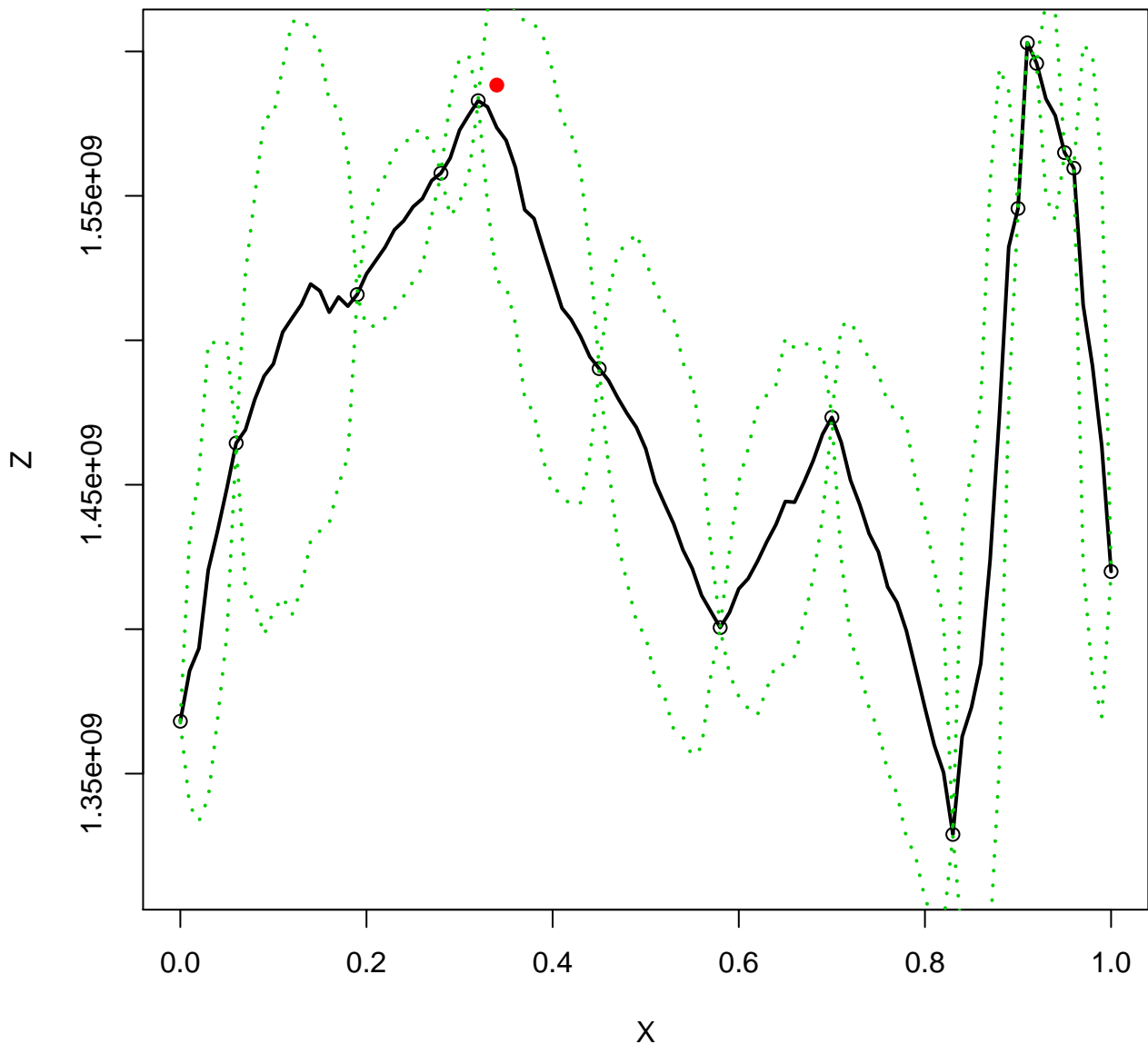
**BART fit (n0 = 10, k = 4)**



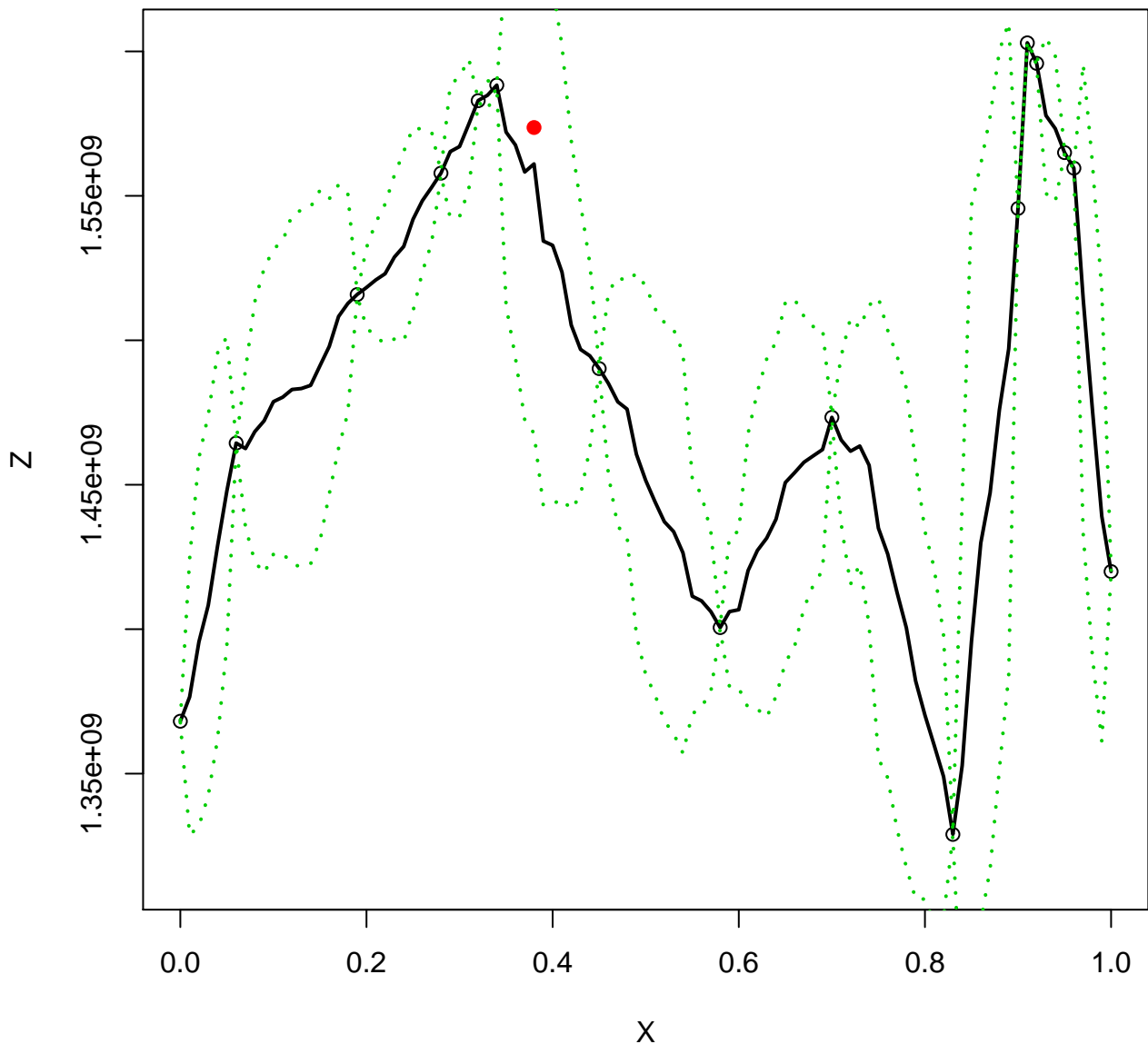
**BART fit (n0 = 10, k = 5)**



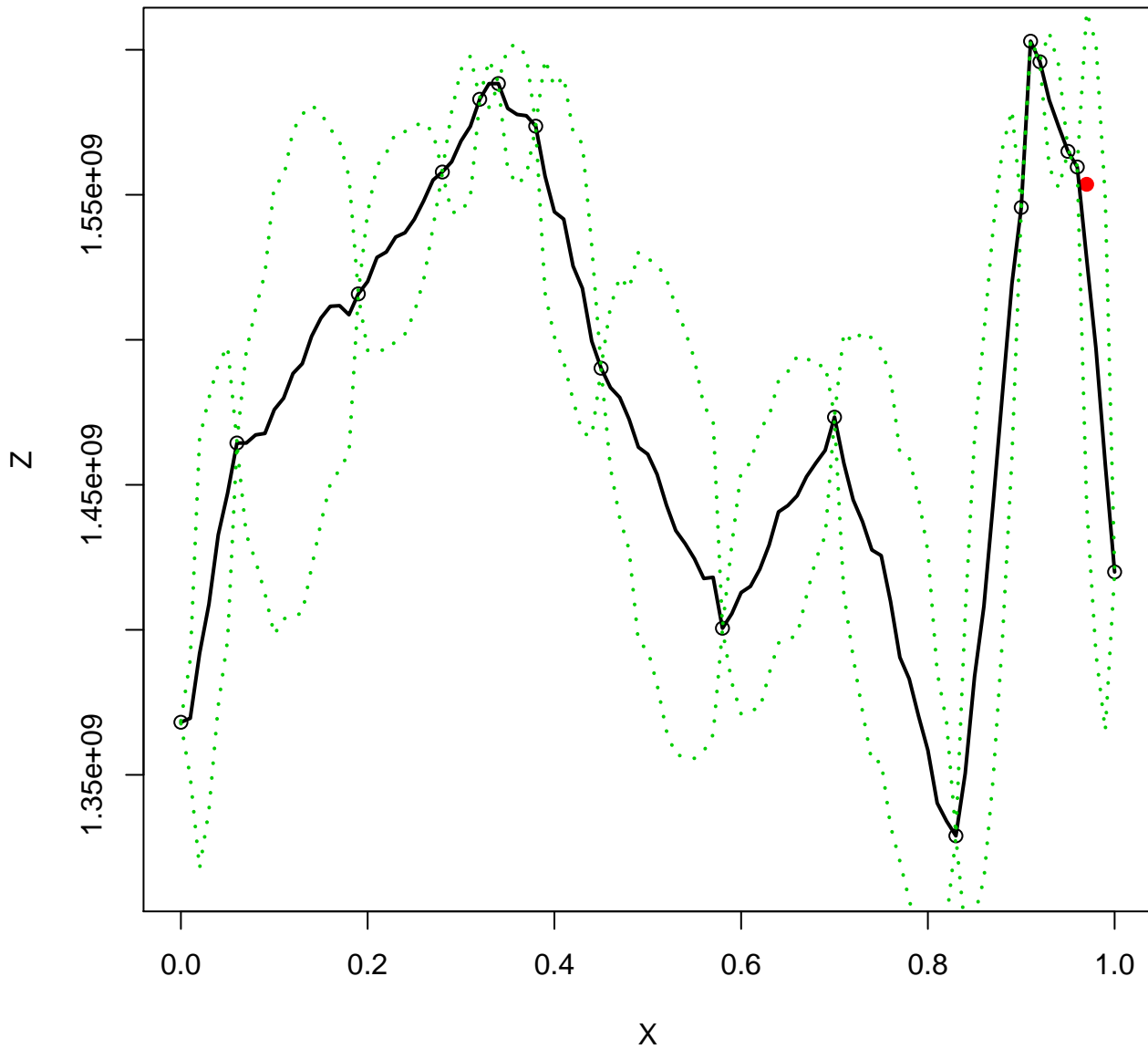
**BART fit (n0 = 10, k = 6)**



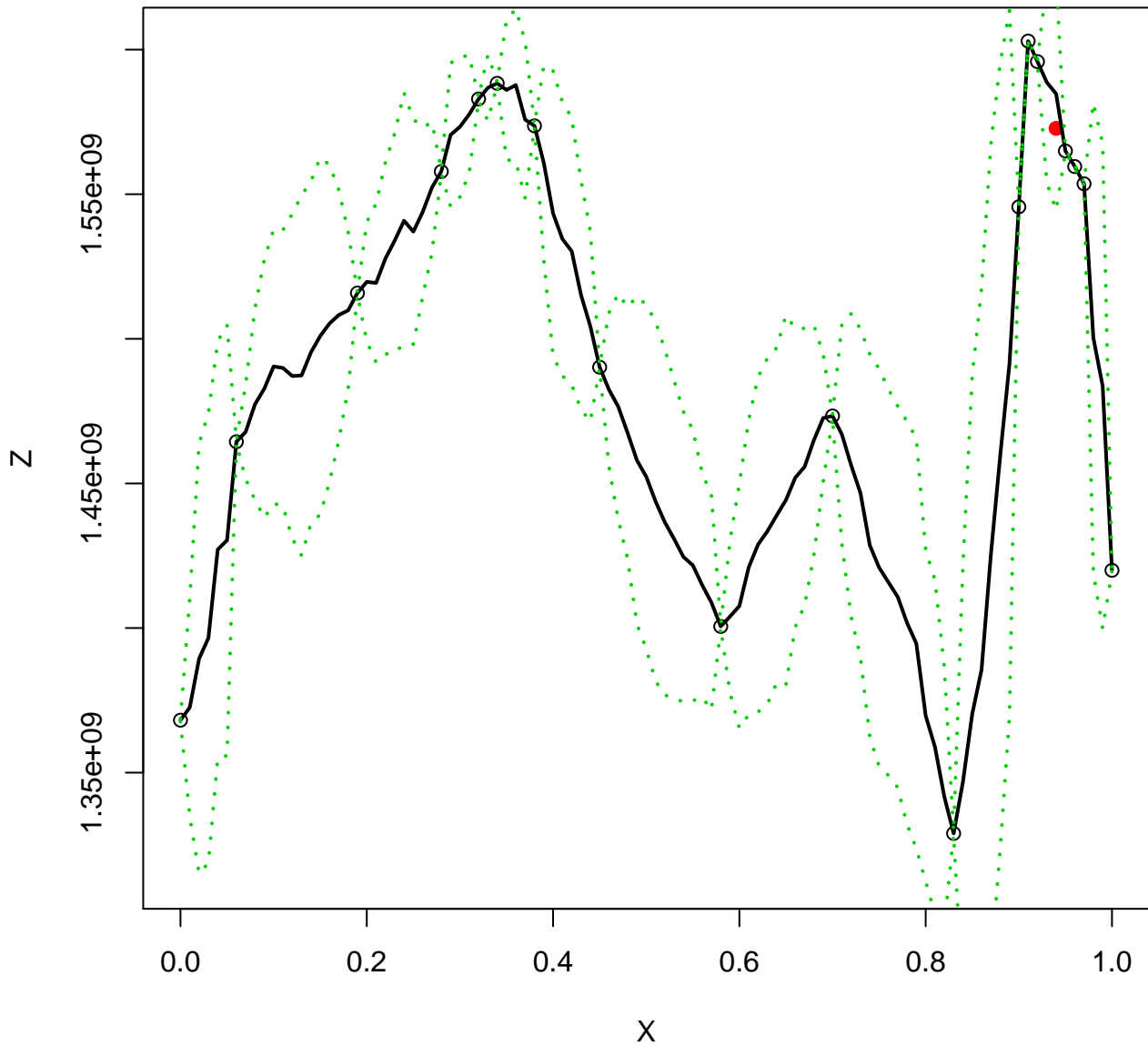
**BART fit (n0 = 10, k = 7)**



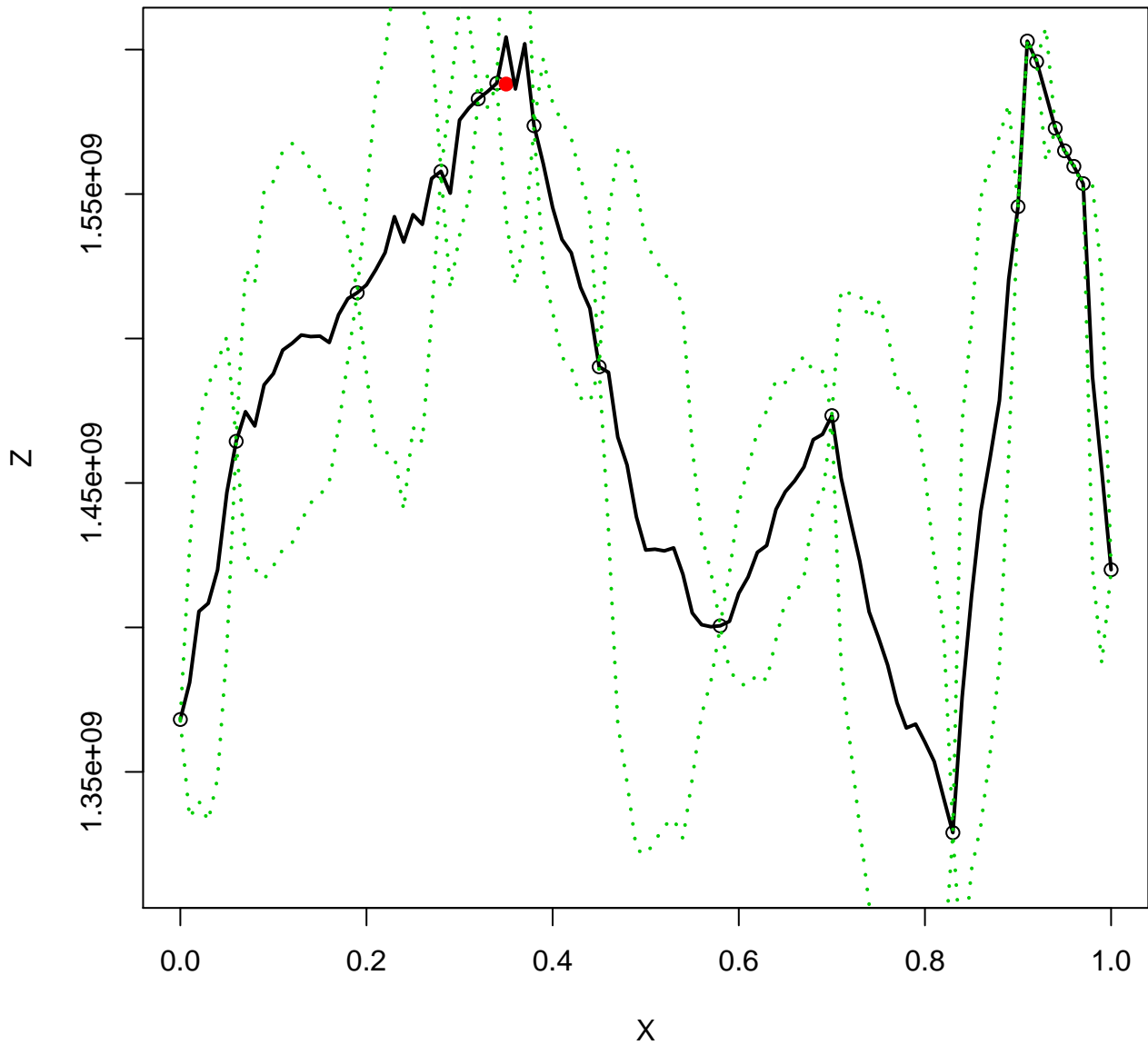
**BART fit (n0 = 10, k = 8)**



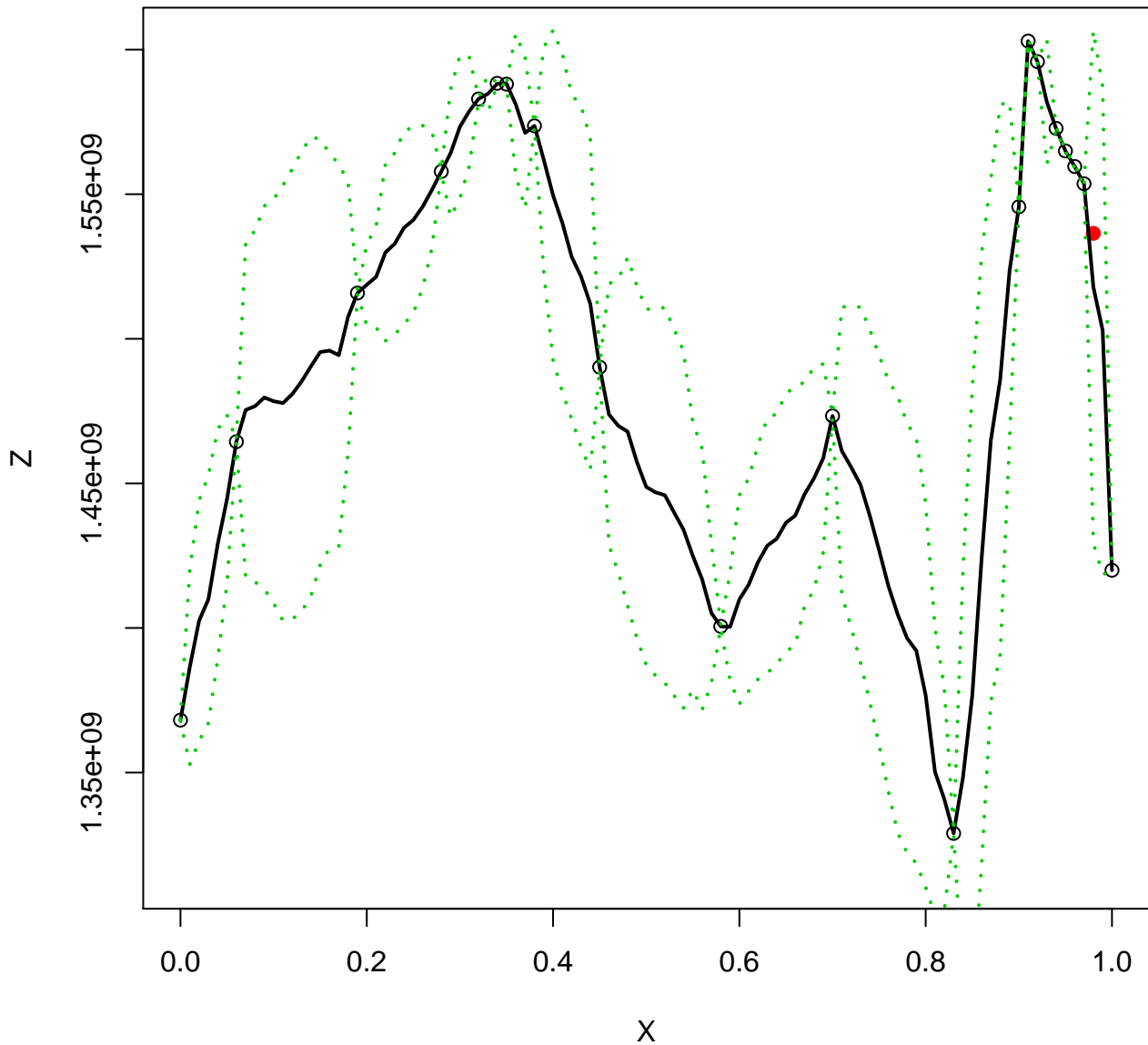
**BART fit (n0 = 10, k = 9)**



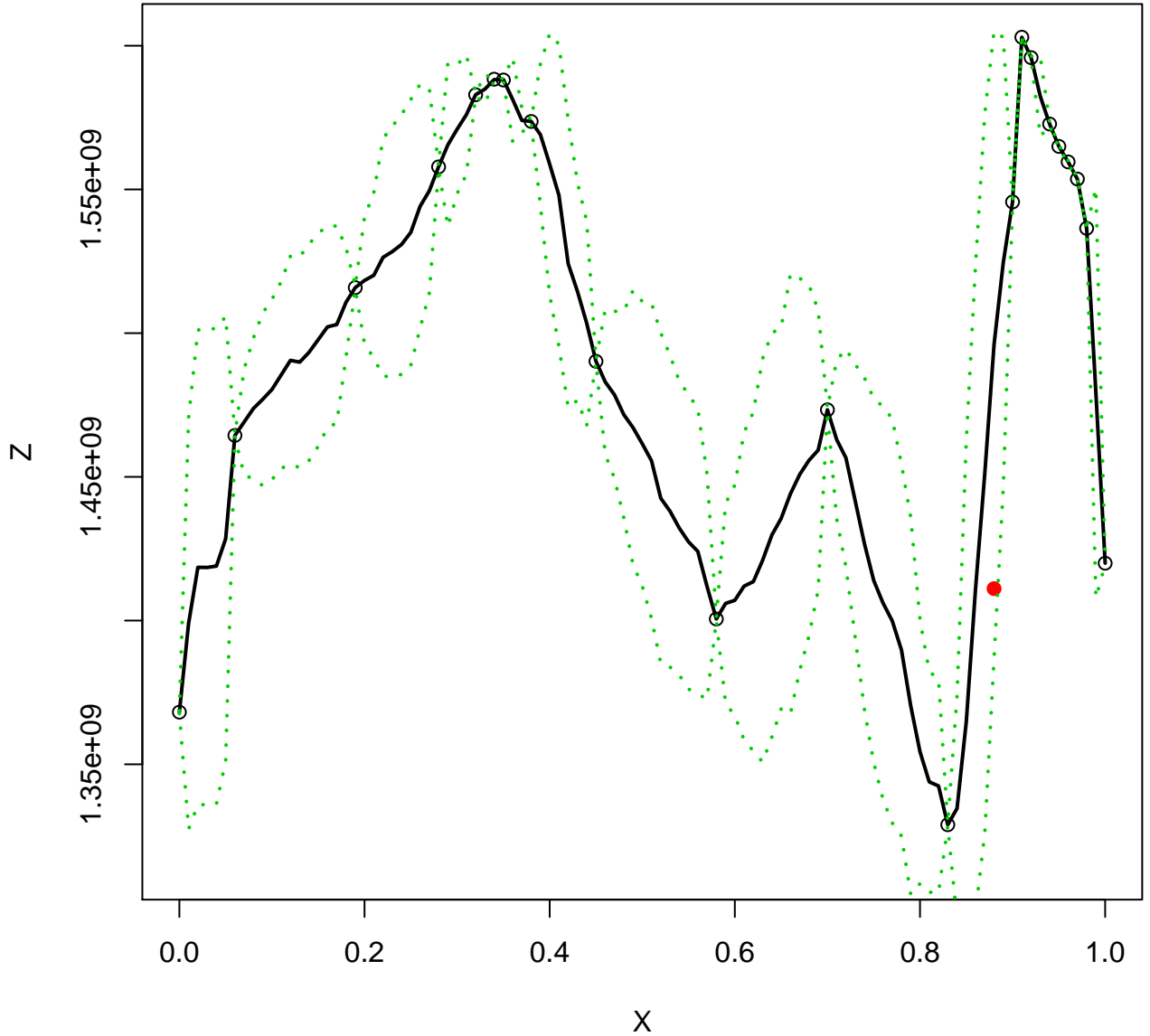
**BART fit (n0 = 10, k = 10)**



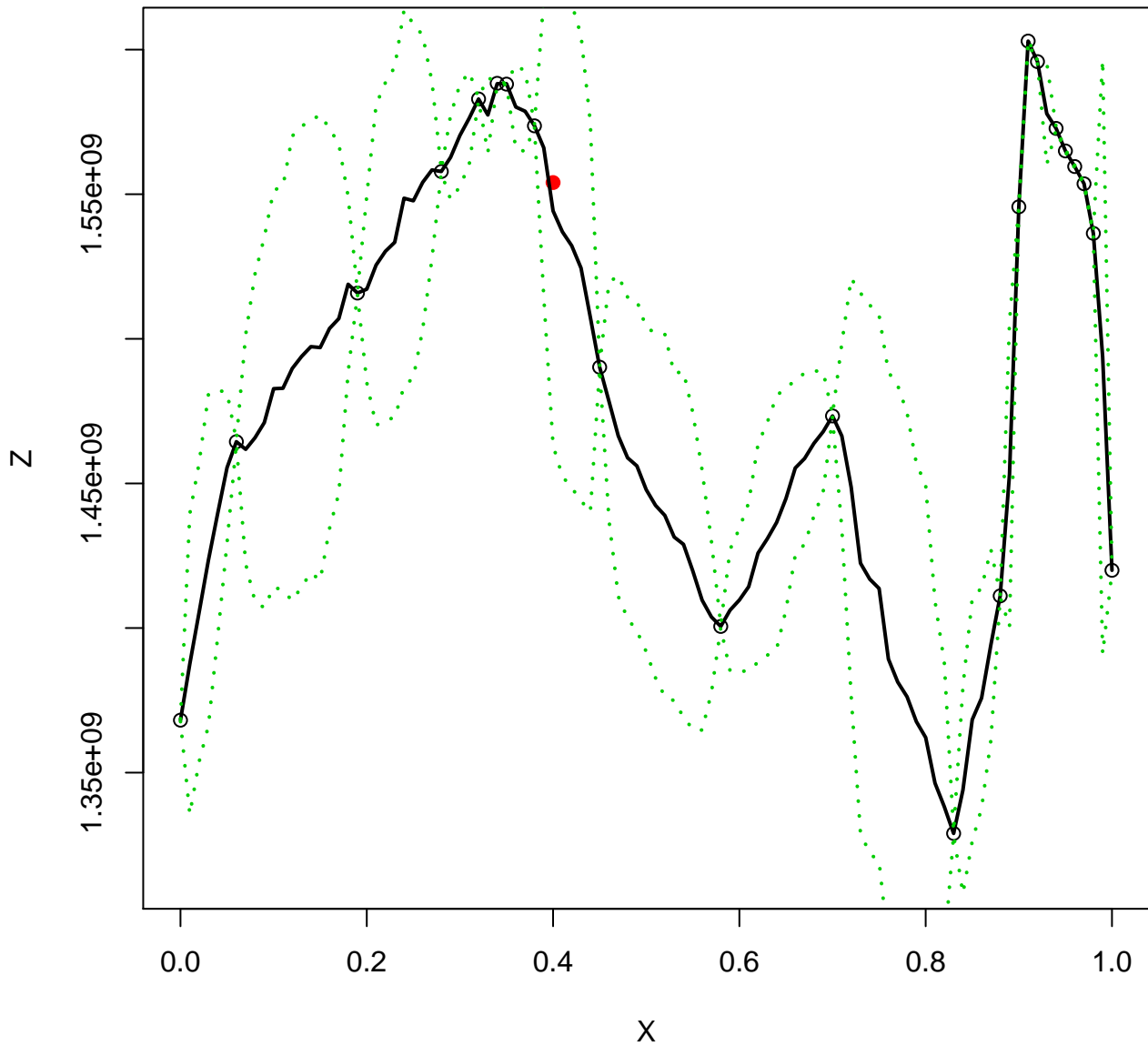
**BART fit (n0 = 10, k = 11)**



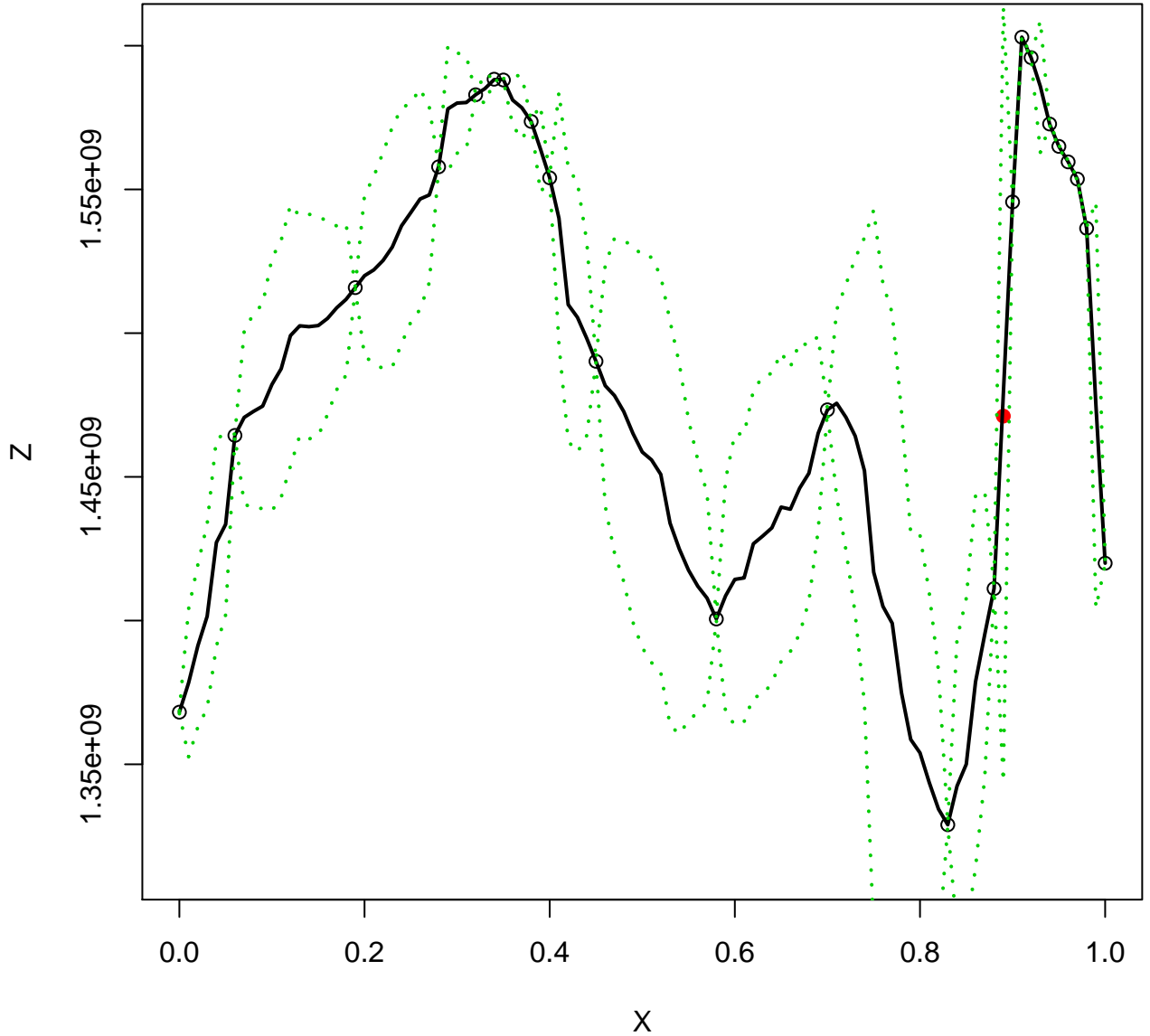
**BART fit (n0 = 10, k = 12)**



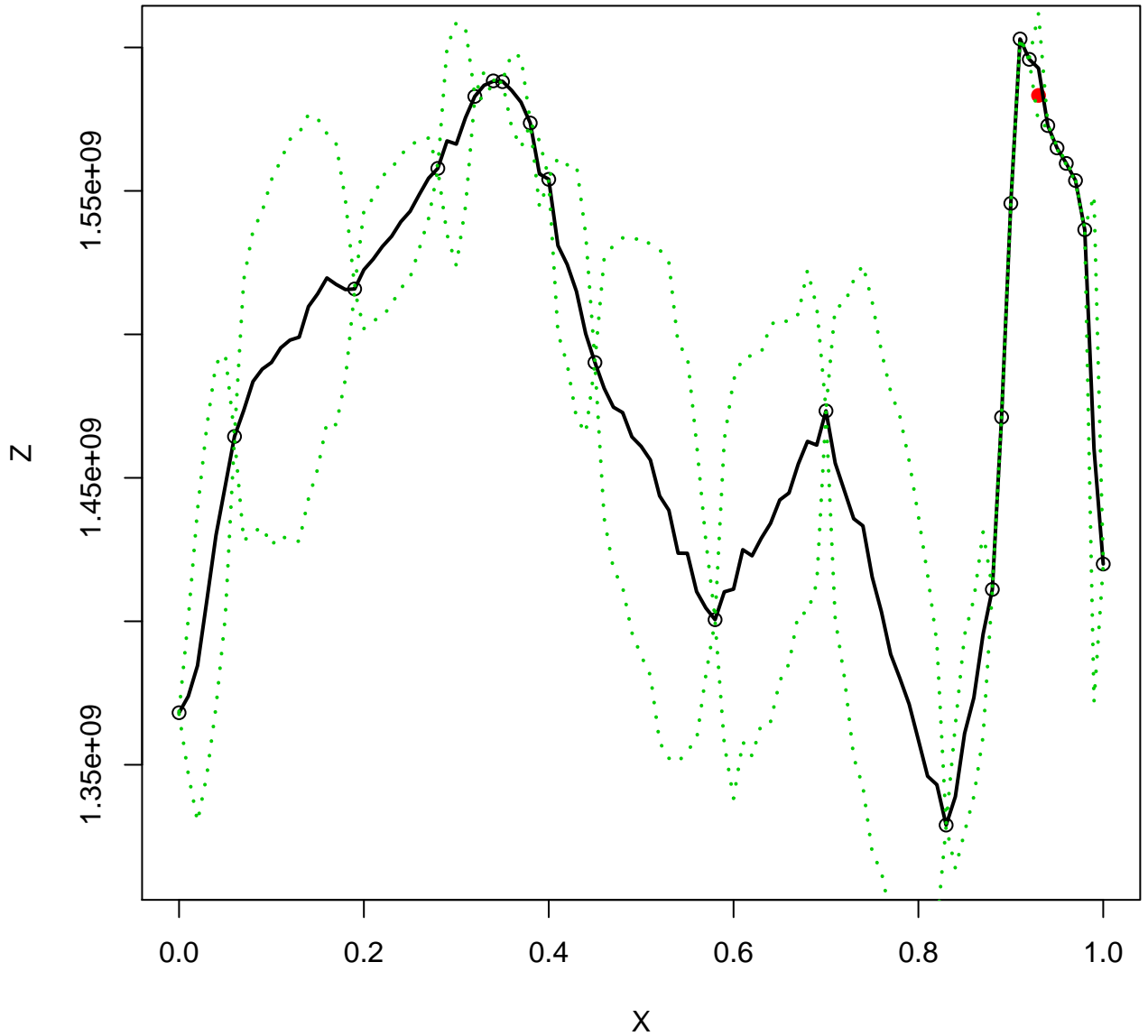
**BART fit (n0 = 10, k = 13)**



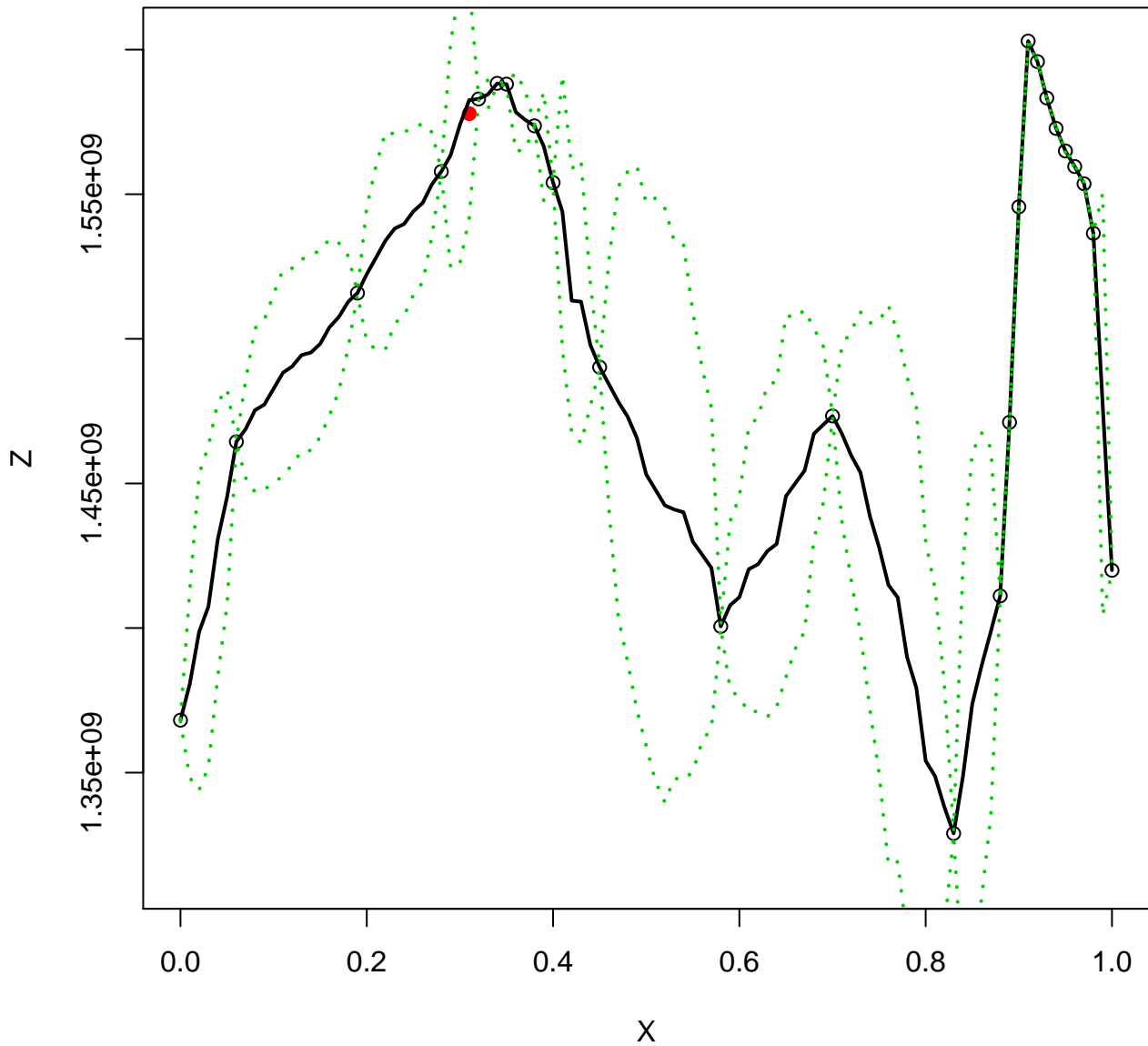
**BART fit (n0 = 10, k = 14)**



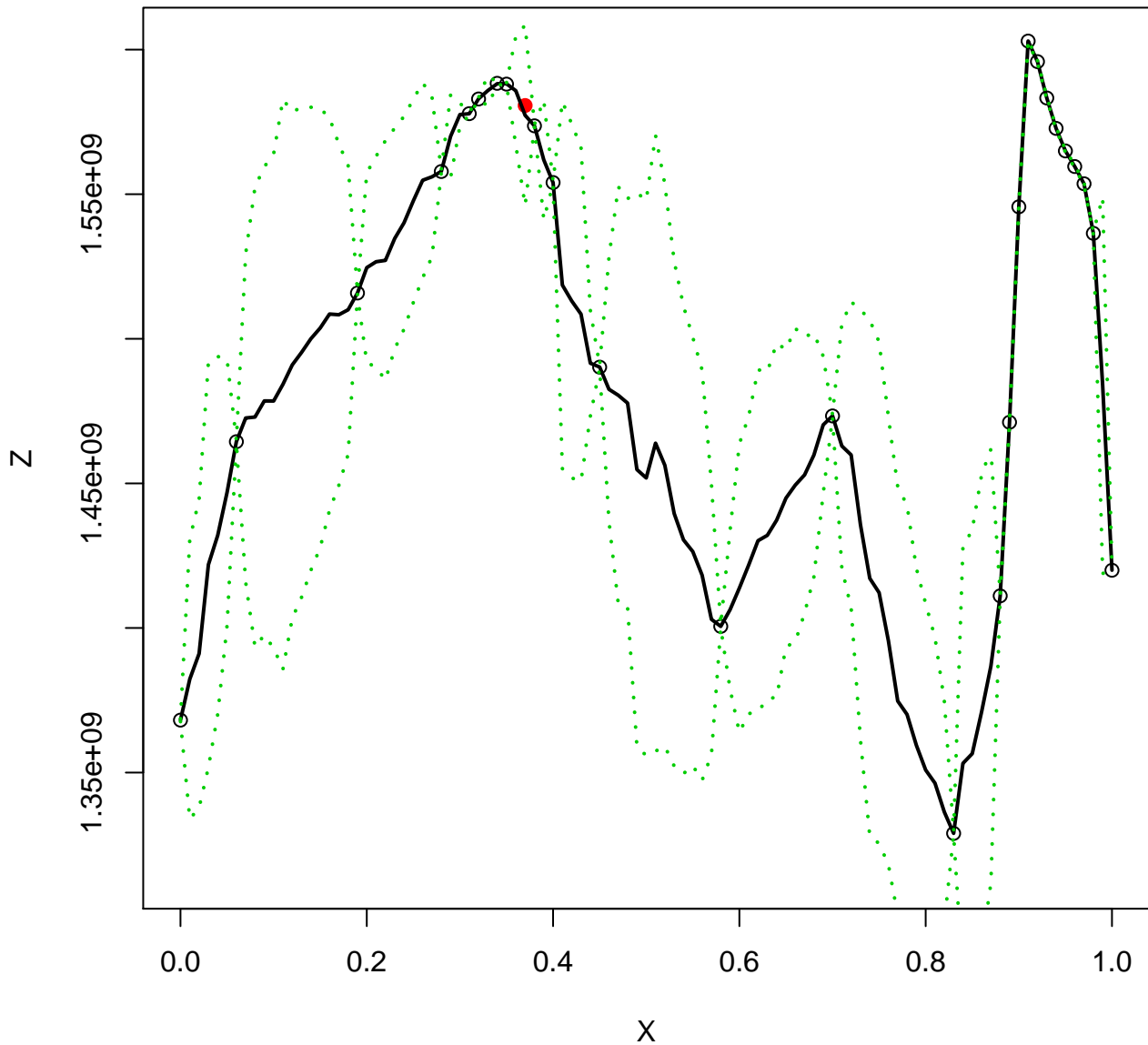
BART fit (n0 = 10, k = 15)



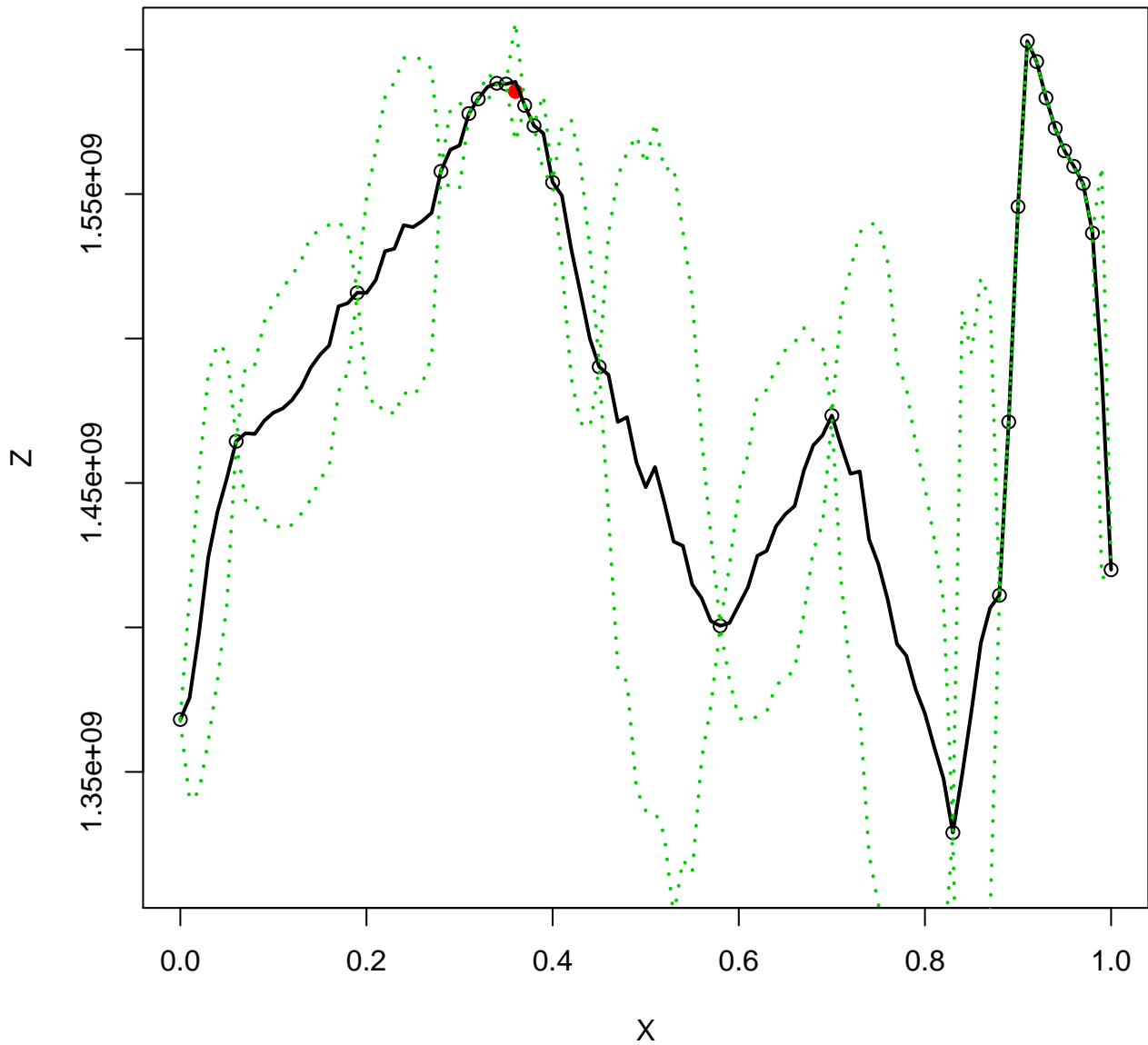
**BART fit (n0 = 10, k = 16)**



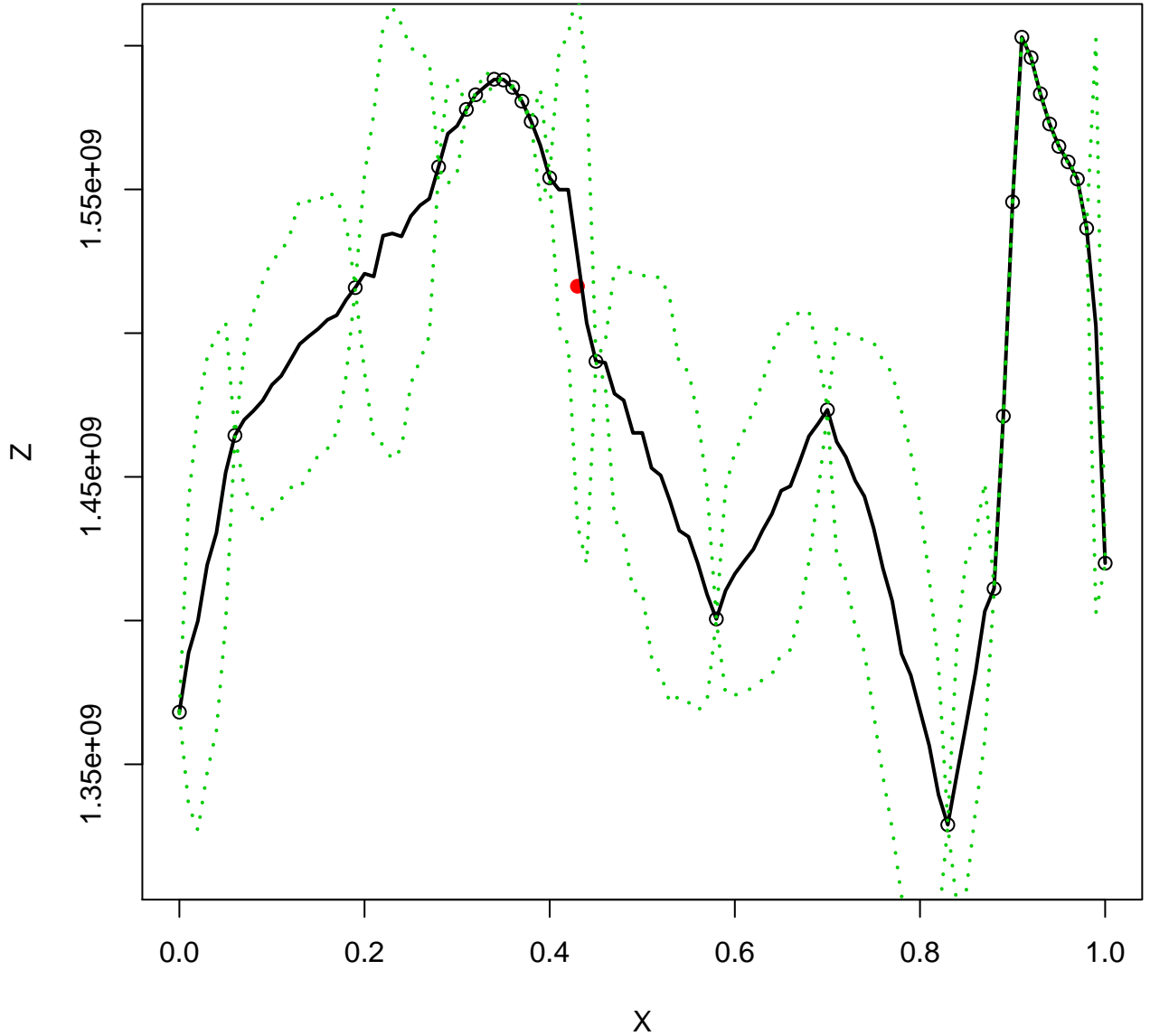
**BART fit (n0 = 10, k = 17)**



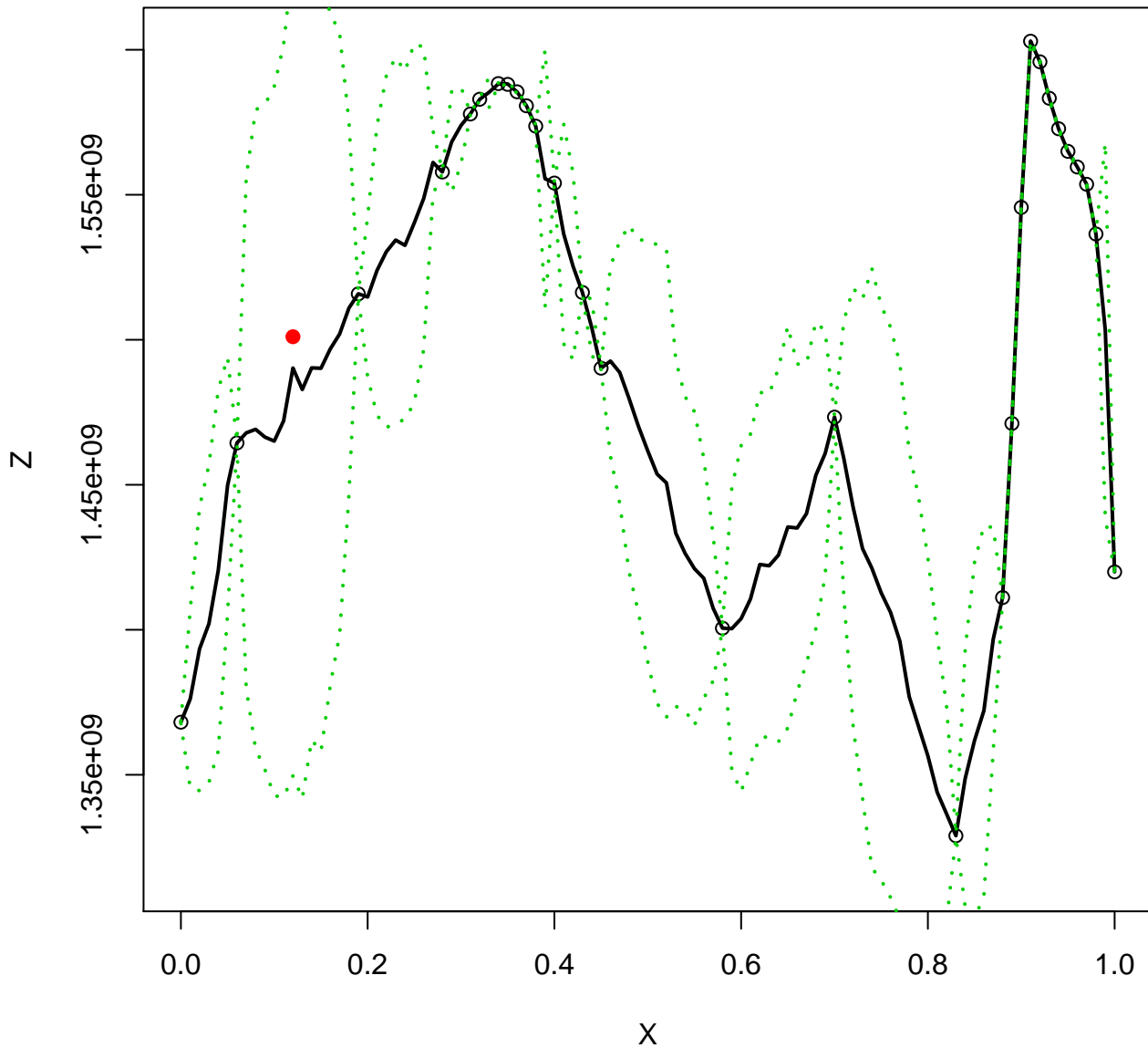
**BART fit (n0 = 10, k = 18)**



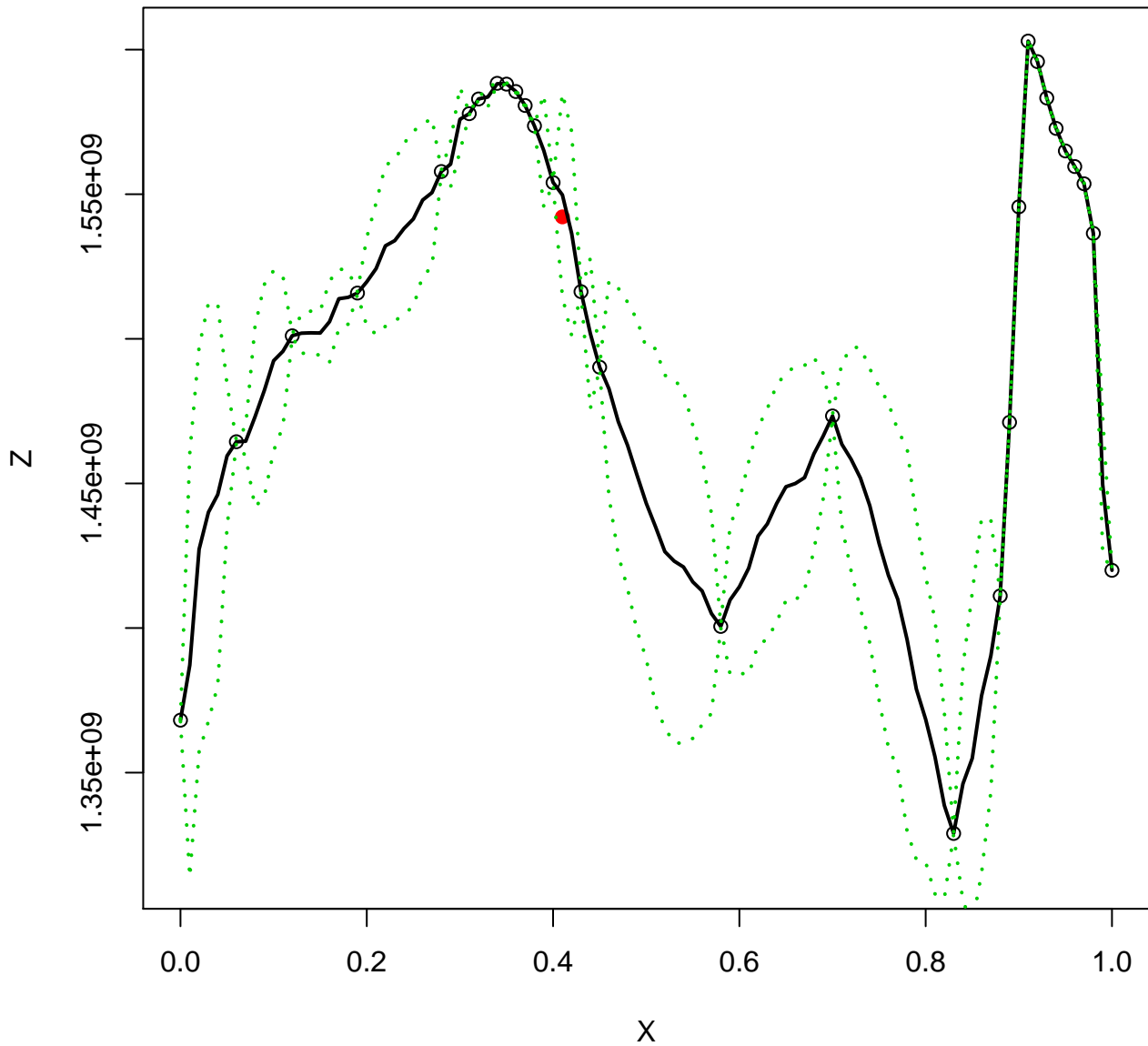
**BART fit (n0 = 10, k = 19)**



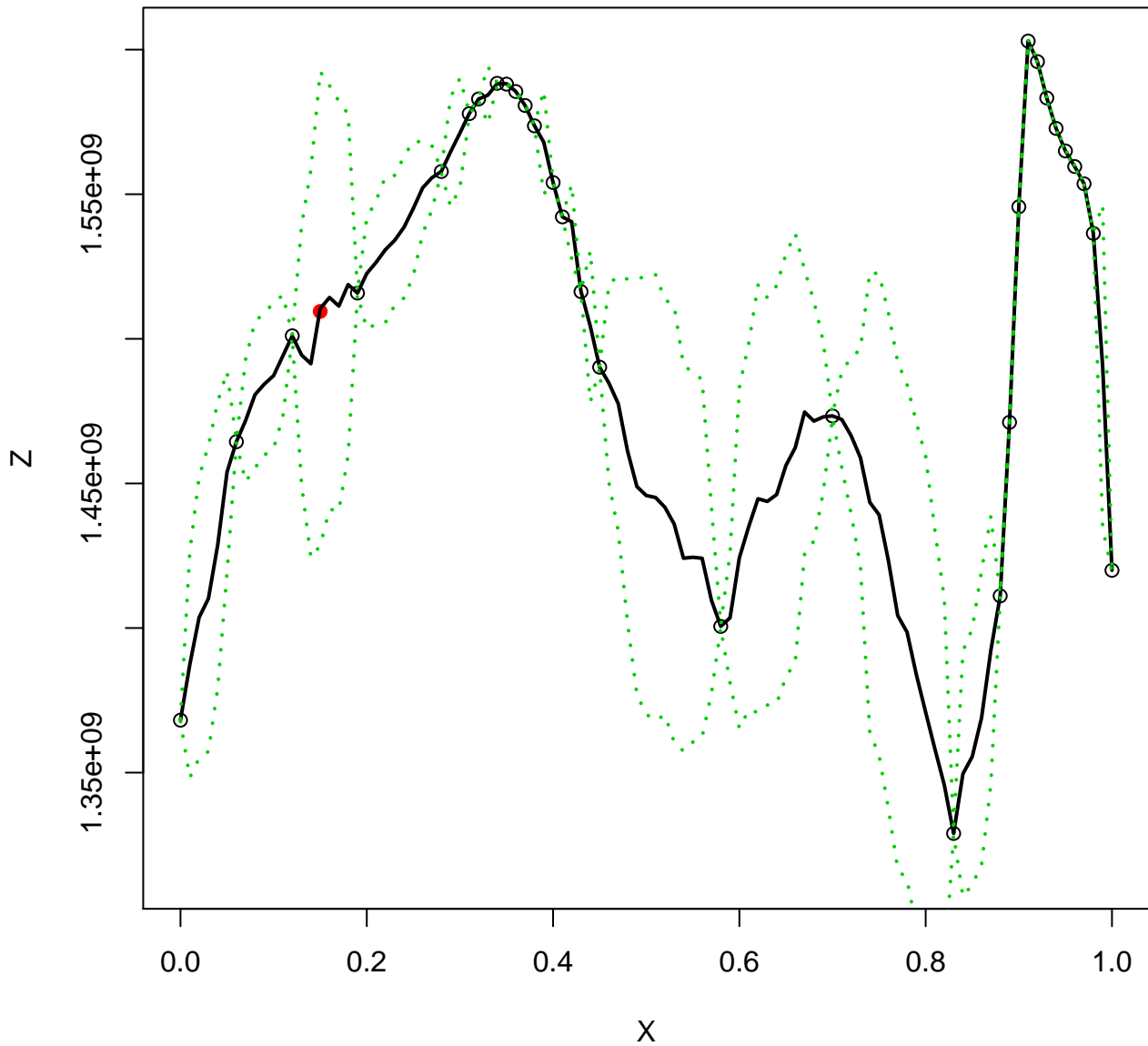
**BART fit (n0 = 10, k = 20)**



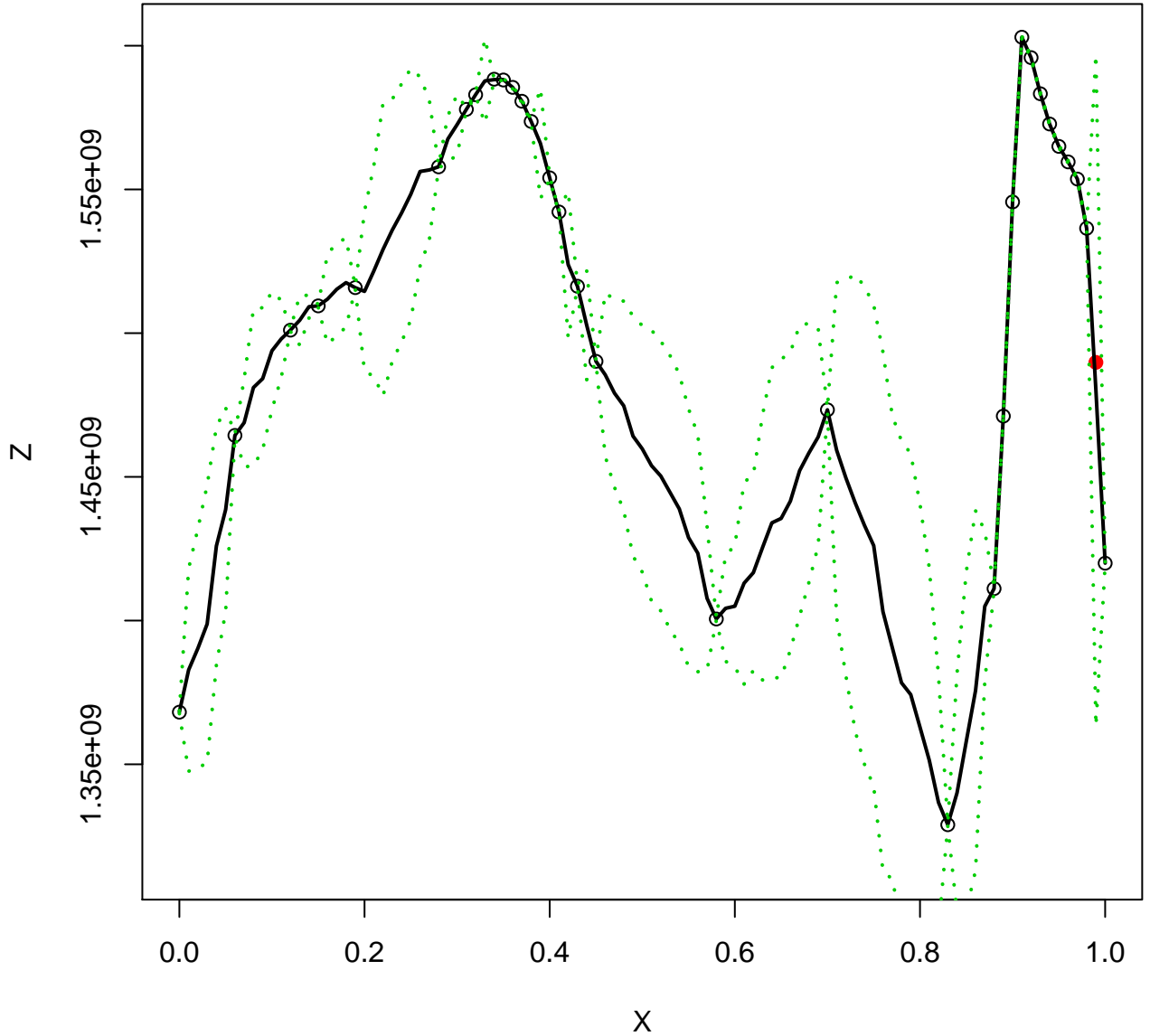
**BART fit (n0 = 10, k = 21)**



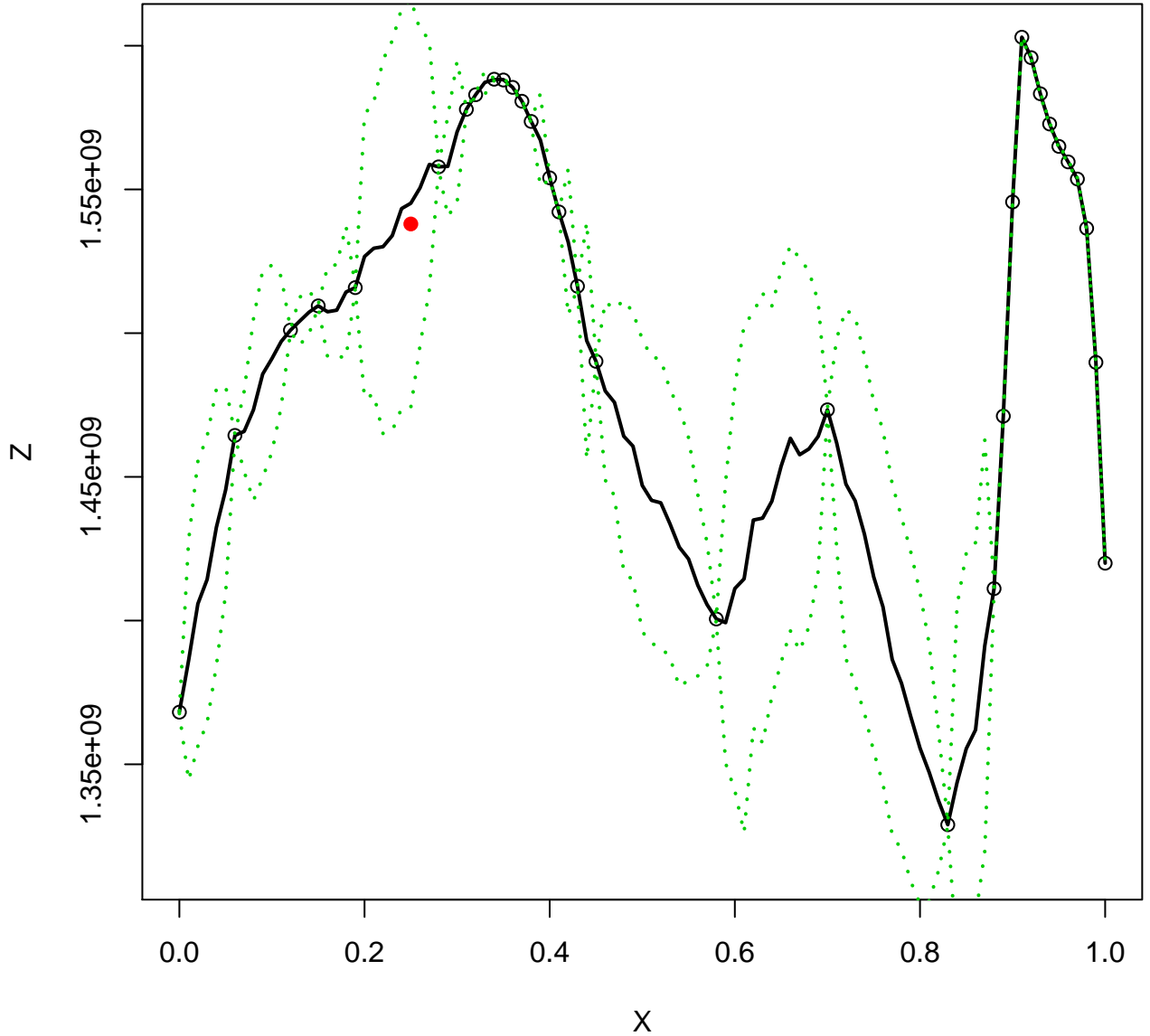
**BART fit (n0 = 10, k = 22)**



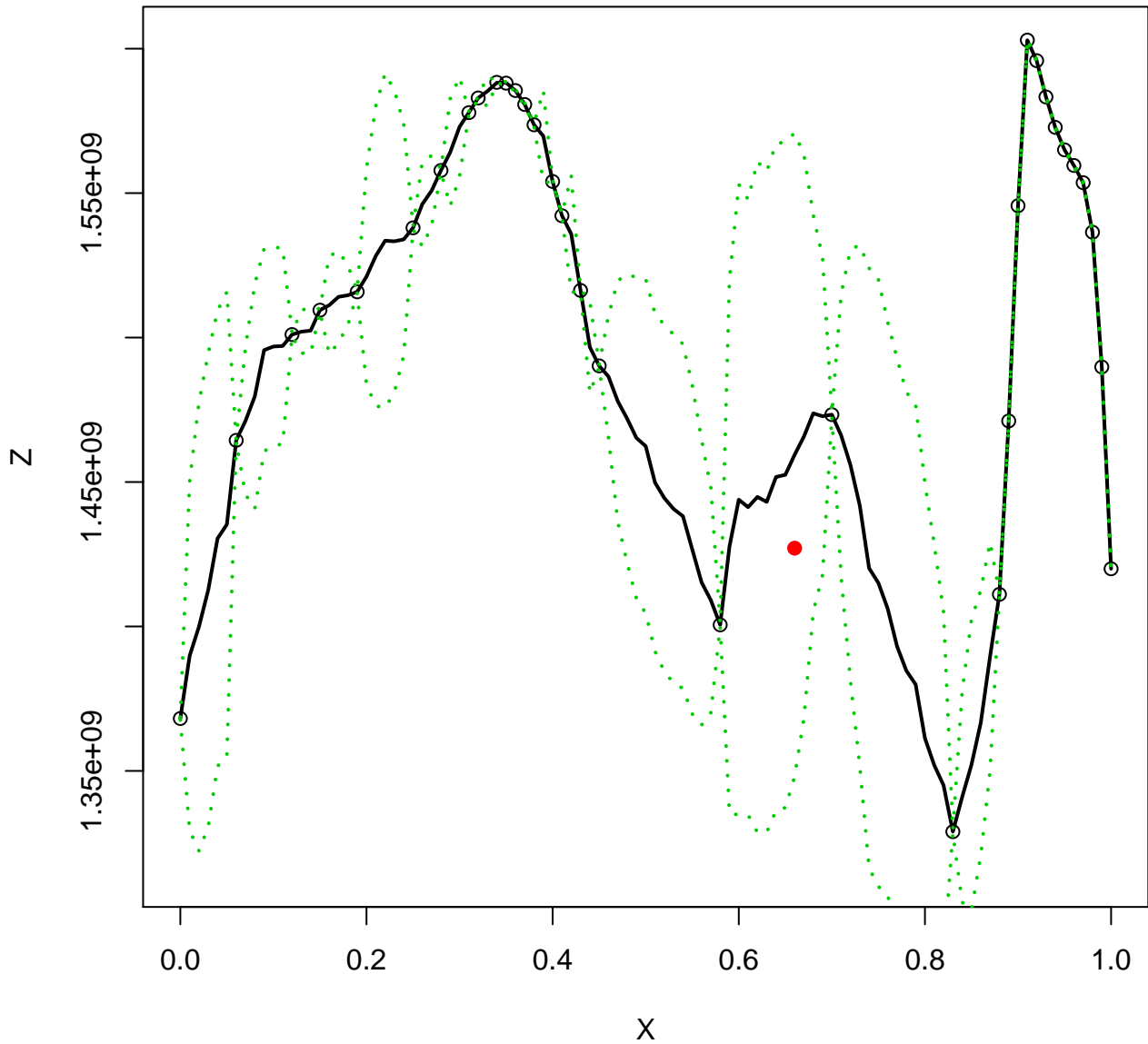
**BART fit (n0 = 10, k = 23)**



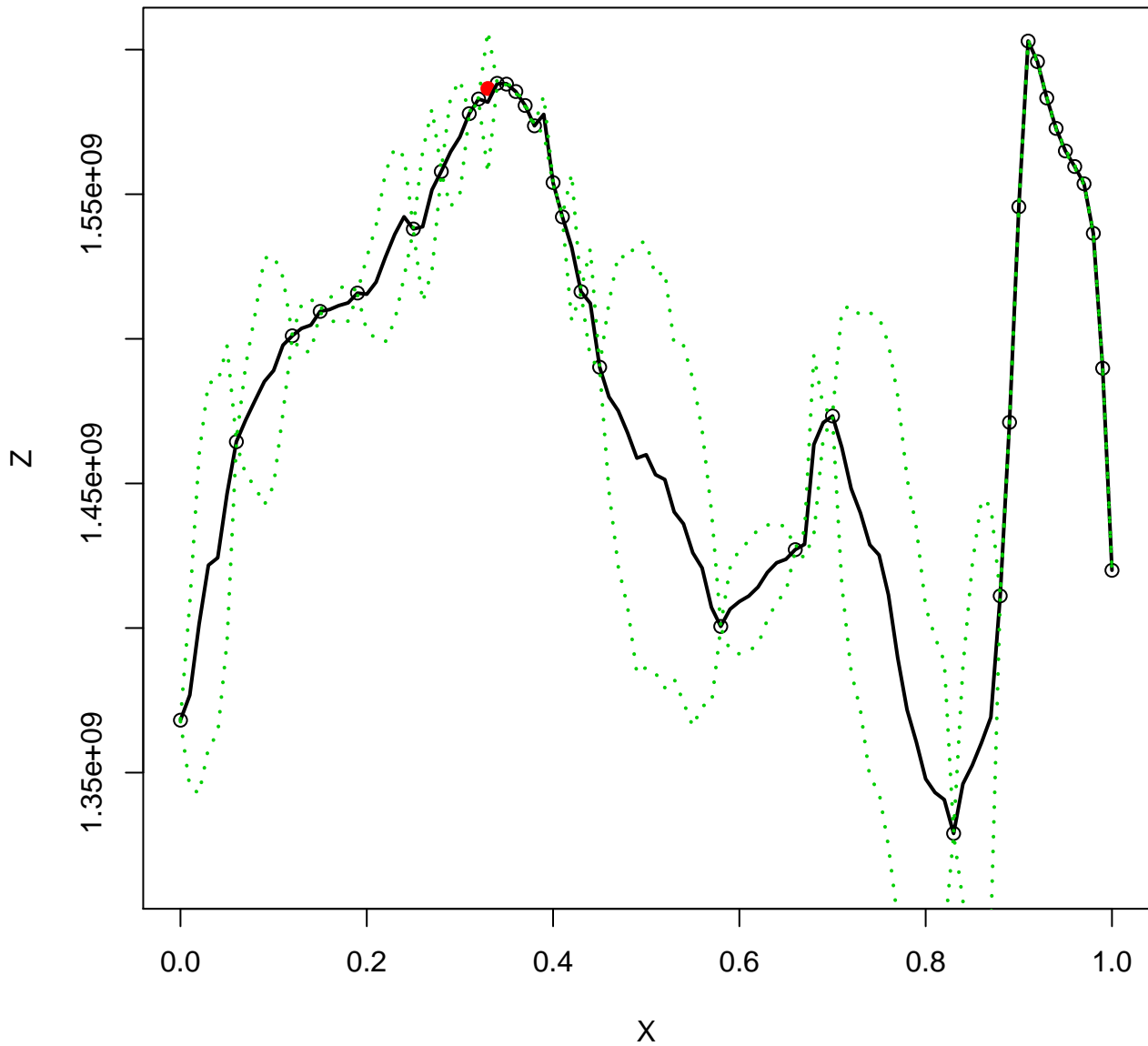
**BART fit (n0 = 10, k = 24)**



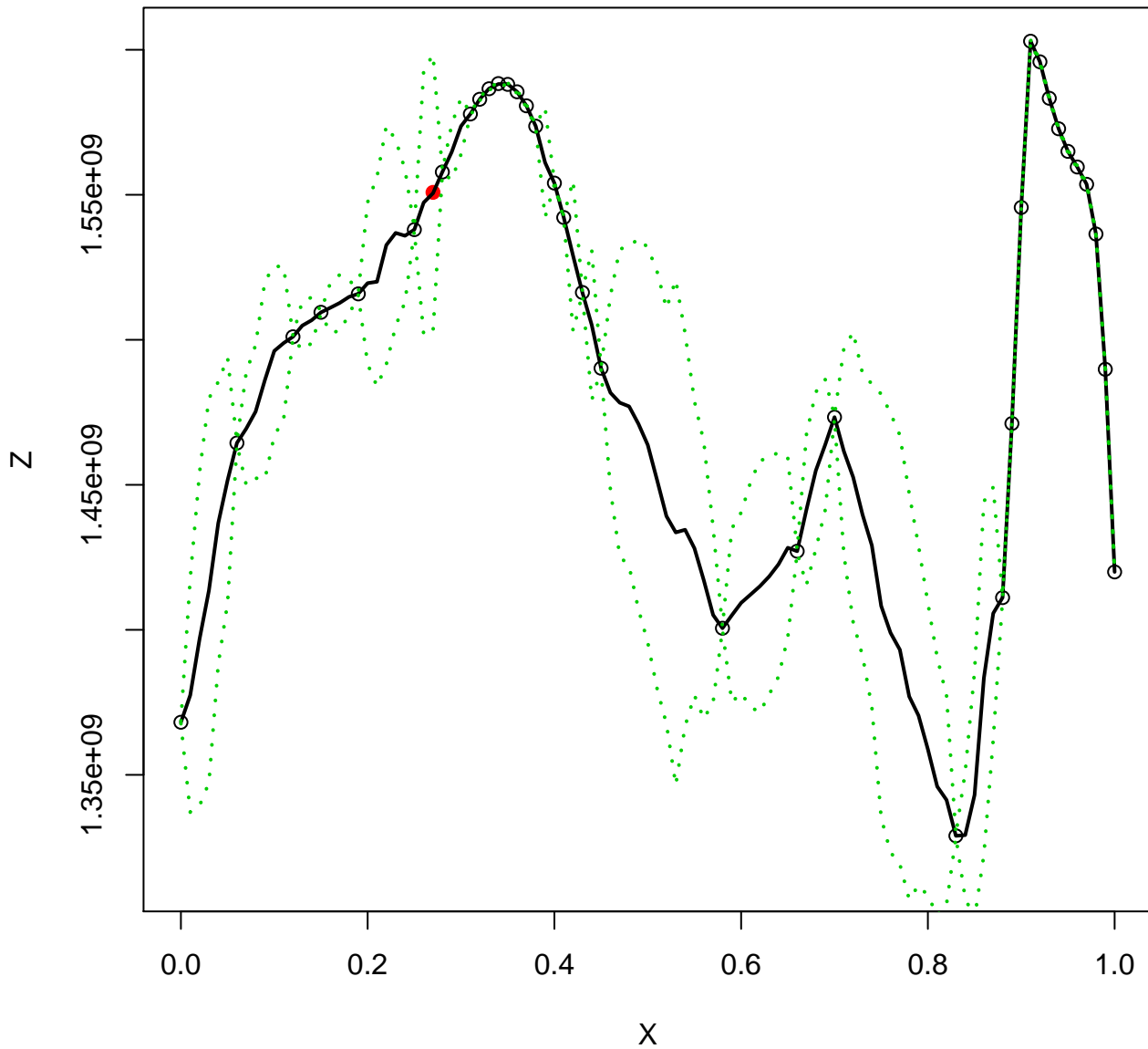
**BART fit (n0 = 10, k = 25)**



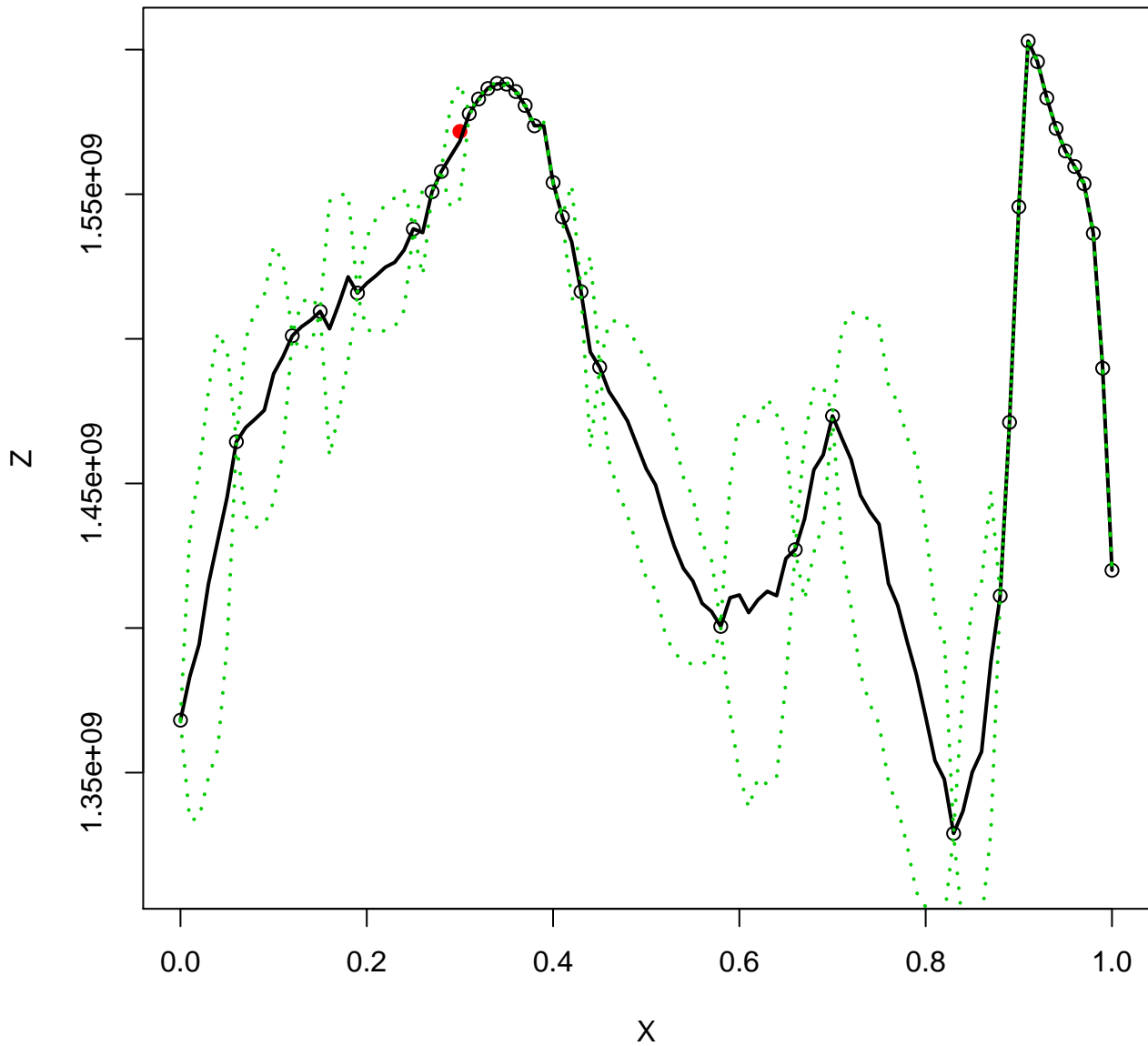
**BART fit (n0 = 10, k = 26)**



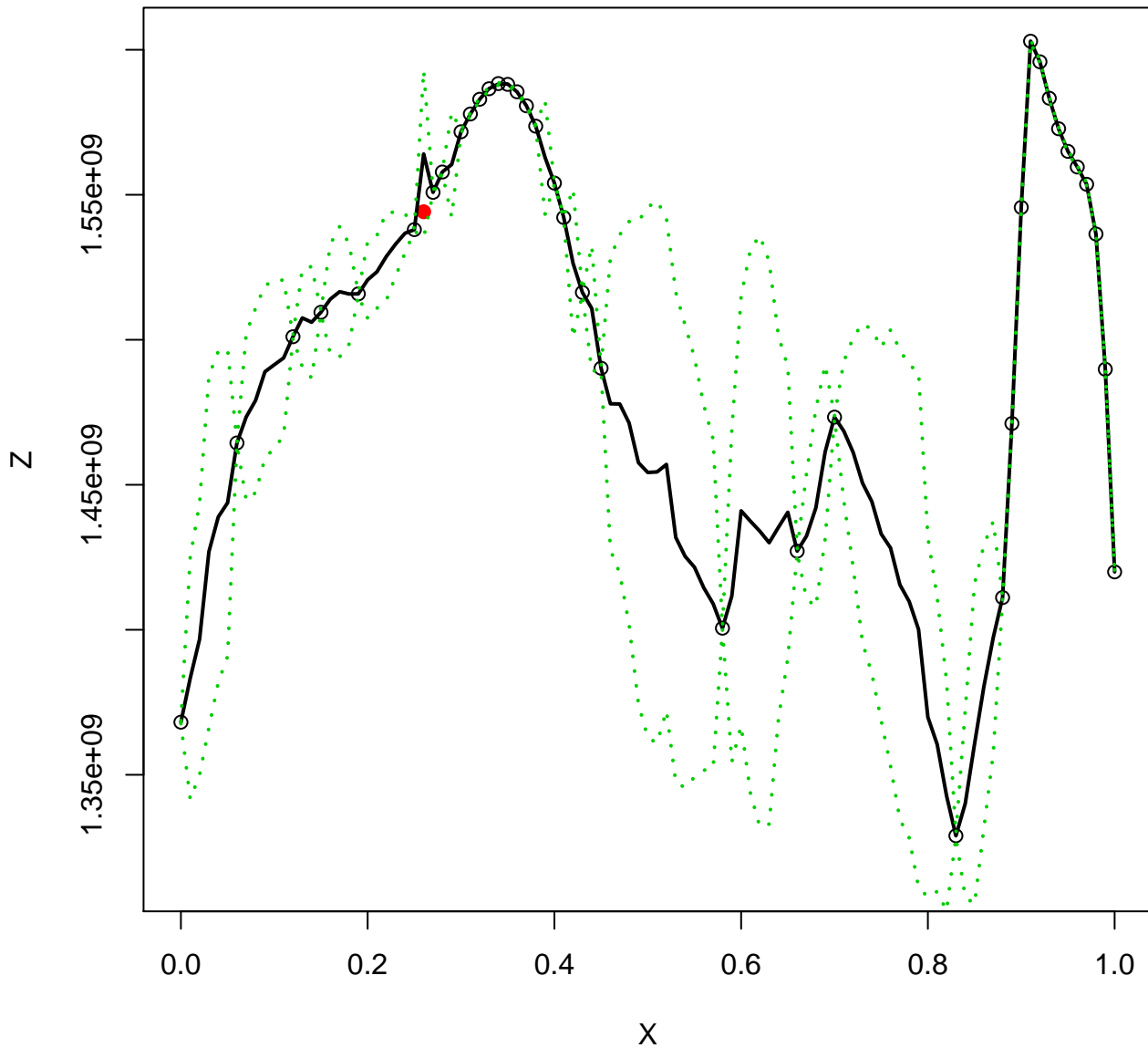
**BART fit (n0 = 10, k = 27)**



**BART fit (n0 = 10, k = 28)**



**BART fit (n0 = 10, k = 29)**



**BART fit (n0 = 10, k = 30)**

