# Dynamic Price Optimization

Prof. Daniel Guetta

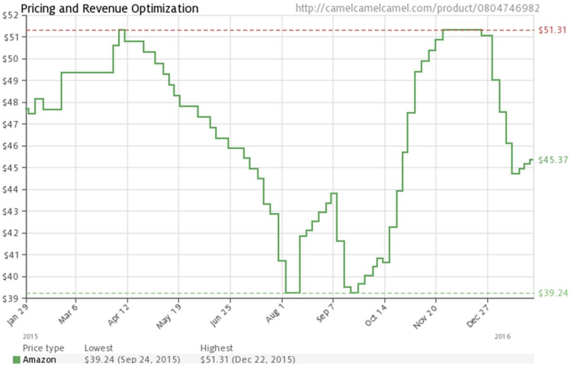
# Price Optimization

Static pricing : one-shot optimization of prices today.

Dynamic pricing : repeatedly optimzing prices day after day.

# Dynamic Pricing

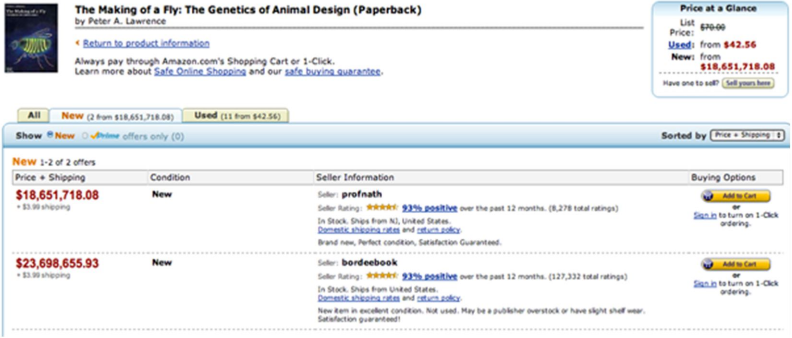
Pricing for "Pricing and Revenue Optimization"
Jan 29 2015 – Jan 29 2016




Why do we observe this pattern of price changes?

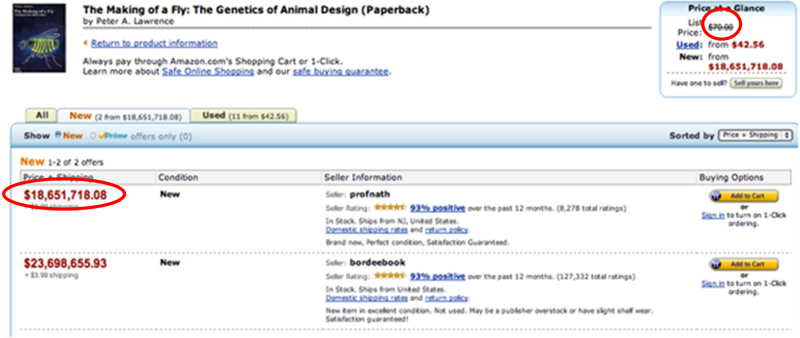
# Dynamic Pricing Gone Wrong

Amazon Webpage – April 18th 2011



# Dynamic Pricing Gone Wrong

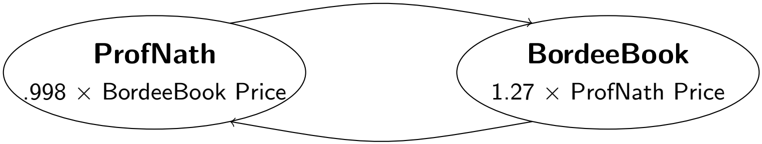
Amazon Webpage – April 18th 2011




Over **$18 million**?! . . . for a book with a list price of $70

# What Happened?

Two competitors watching each other, and setting their price solely on the other's price:



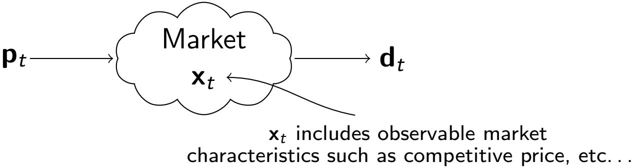

| ProfNath Price | BordeeBook Price |
|---|---|
| \$70.00 | \$88.90 |
| \$88.75 | \$112.71 |
| \$112.52 | \$142.90 |
| \$142.66 | \$181.17 |
| ... | ... |

# Dynamic Price Optimization

We saw, earlier, that the optimal price is given by

$$\epsilon(p^\star) = p^\star/(p^\star - c)$$

This is all nice and well, but what if we don't know $\epsilon(p)$?




Given a sequence of observations $(\mathbf{p}_1, \mathbf{x}_1, \mathbf{d}_1), \cdots, (\mathbf{p}_t, \mathbf{x}_t, \mathbf{d}_t)$ and observing $\mathbf{x}_{t+1}$, what price $\mathbf{p}_{t+1}$ should we choose to maximize expected profitability?

# Outline

1. Parametric Methods

2. An optimization-based approach

3. A Global, gradient-free approach

# Outline

1. Parametric Methods
2. An optimization-based approach
3. A Global, gradient-free approach

# Parametric methods

- Appropriate for known price-response curves.
- Use past data to estimate the parameters of these price-response curves

# Parametric methods

1. Choose a price-response function form (logit, probit, etc...)

# Parametric methods

1. Choose a price-response function form (logit, probit, etc...)
2. Estimate the parameters using past observations

# Parametric methods

1. Choose a price-response function form (logit, probit, etc...)
2. Estimate the parameters using past observations
3. Find the price that satisfies the optimality conditions given the current value of the parameters:

$$\epsilon(p^\star) = p^\star/(p^\star - c)$$

# Parametric methods

1. Choose a price-response function form (logit, probit, etc...)
2. Estimate the parameters using past observations
3. Find the price that satisfies the optimality conditions given the current value of the parameters:

$$\epsilon(p^\star) = p^\star / (p^\star - c)$$

4. Set that price and observe demand. Return to step 2

# Parametric methods

1. Choose a price-response function form (logit, probit, etc...)
2. Estimate the parameters using past observations
3. Find the price that satisfies the optimality conditions given the current value of the parameters:

$$\epsilon(p^{\star}) = p^{\star}/(p^{\star} - c)$$

4. Set that price and observe demand. Return to step 2

Parameter estimates “should” converge to reasonable estimates of the true price-response function.

# Parametric Approach – Example

- Suppose products cost \$50 per unit, and assume the weekly price-response curve is linear; $d(p) = D - bp$.
- Suppose that from past data, we know that at prices $p_a = \$75$ and $p_b = \$65$, demand was $d(p_a) = 8,900$ and $d(p_b) = 11,046$ respectively.
- Use a linear regression and this past data to estimate $D_0 = 24,955$ and $b_0 = 214.60$.

# Parametric Approach – Example

- Suppose products cost \$50 per unit, and assume the weekly price-response curve is linear; $d(p) = D - bp$.
- Suppose that from past data, we know that at prices $p_a = \$75$ and $p_b = \$65$, demand was $d(p_a) = 8,900$ and $d(p_b) = 11,046$ respectively.
- Use a linear regression and this past data to estimate $D_0 = 24,955$ and $b_0 = 214.60$.
- Using static optimization with this estimated price-response curve, we get an optimal price of $p_1 = \$83.24$. You implement this price and, a week later, observe a demand of $d_1(p_1) = 7,865$.

# Parametric Approach – Example

- Suppose products cost \$50 per unit, and assume the weekly price-response curve is linear; $d(p) = D - bp$.
- Suppose that from past data, we know that at prices $p_a = \$75$ and $p_b = \$65$, demand was $d(p_a) = 8,900$ and $d(p_b) = 11,046$ respectively.
- Use a linear regression and this past data to estimate $D_0 = 24,955$ and $b_0 = 214.60$.
- Using static optimization with this estimated price-response curve, we get an optimal price of $p_1 = \$83.24$. You implement this price and, a week later, observe a demand of $d_1(p_1) = 7,865$.
- Use the last three datapoints and linear regression to estimate $D_1 = 23,995$ and $b_2 = 174.43$. Put the corresponding optimal price of $p_4 = \$89.08$ into the market and repeat.

# Parametric Approach – The Problem

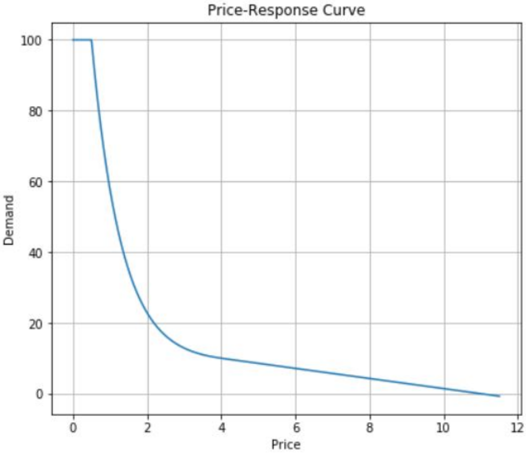
- The assumed form of the price-response curve could be incorrect.

# Parametric Approach – The Problem

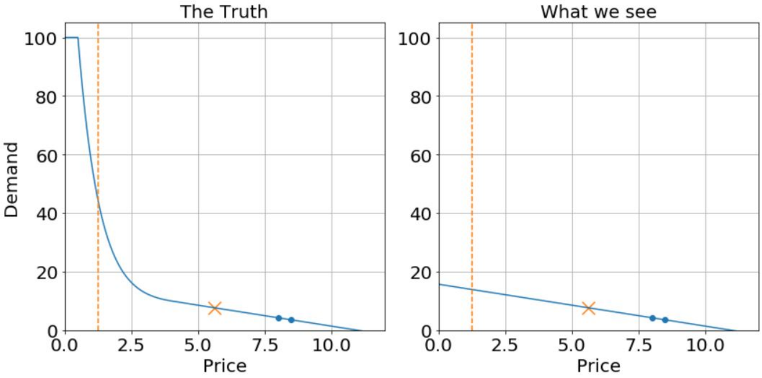
- The assumed form of the price-response curve could be incorrect.
- 'Optimal' values at every step $\Rightarrow$ the prices may 'jam' at a sub-optimal point.

# An Example

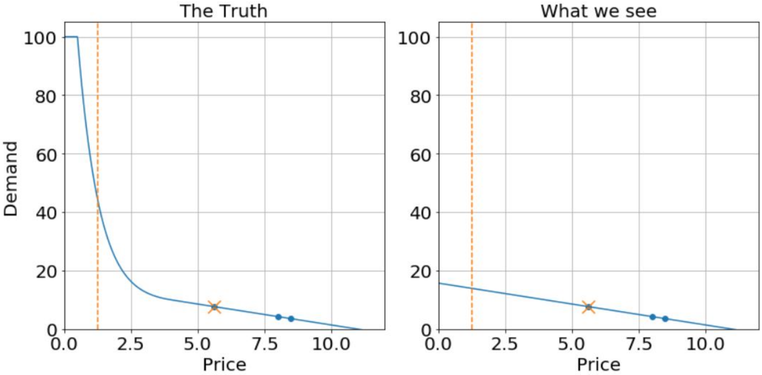
Suppose this is our price-response function, and that the unit price to produce an item is $c = \$0.25$.



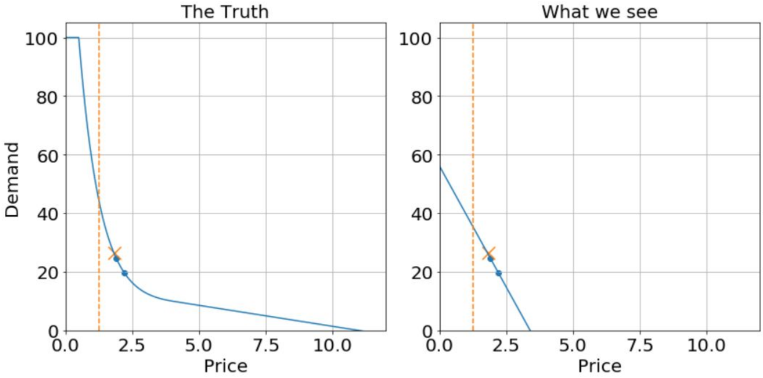
# Starting at \$8 and \$8.5



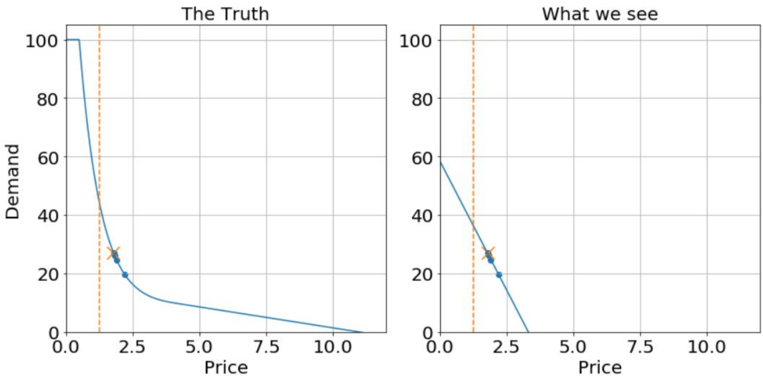
# Starting at $8 and $8.5



# Starting at \$1.9 and \$2.2



# Starting at $1.9 and $2.2



# Outline

1. Parametric Methods
2. An optimization-based approach
3. A Global, gradient-free approach

# An optimization-based approach

Fundamentally, our aim is to maximize the profit function

$$\Pi(p) = (p - c)d(p)$$

How do we get to $\Pi'(p) = 0$?

# An optimization-based approach

Fundamentally, our aim is to maximize the profit function

$$\Pi(p) = (p - c)d(p)$$

How do we get to $\Pi'(p) = 0$?

We can use classical gradient-descent methods to optimize this function.

To keep in line with the literature, we'll couch everything in terms of *minimization*.

# An optimization-based approach

Pick a parameter $\delta$ and a sequences of parameters $\{\sigma_t\}$.

# An optimization-based approach

Pick a parameter $\delta$ and a sequences of parameters $\{\sigma_t\}$.

1. Estimate derivative of $\Pi$ around current point

$$\Pi'(p_t) \approx \frac{\Pi(p_t + \delta) - \Pi(p_t)}{\delta}$$

2. Set

$$p_{t+1} = p_t - \sigma_t \Pi'(p_t)$$

# Does it work?

Suppose $\Pi(p) = p^2$, and that we start at $p_0 = 10$.

- With $\sigma_t = 0.8$, the sequence of $p$'s goes

$$-6, 3.6, -2.16, 1.3, -0.78, 0.47, -0.28, 0.17, \cdots$$

- With $\sigma_t = 1.5$, the sequence of $p$'s goes

$$-20, 40, -80, 160, -320, 640, -1280, 2560, \cdots$$

- With $\sigma_t = 10^{-3}$, the sequence of $p$'s goes

$$9.98, 9.96, 9.94, 9.92, 9.9, 9.88, 9.86, 9.84, \cdots$$

- With $\sigma_t = 5/t$, the sequence of $p$'s goes

$$-90, 360, -840, 1260, -1260, 840, -360, 90, -10, 0, \cdots$$

# Proof of convergence (Optional)

From Taylor's Theorem, we have that for any $p$ and $\bar{p}$

$$\Pi(\bar{p}) = \Pi(p) + \Pi'(p)(\bar{p} - p) + \frac{1}{2}\Pi''(z)(\bar{p} - p)^2$$

for some $z \in [p, \bar{p}]$.

Let us now assume that the second derivative of $\Pi$ is bounded by some constant $L$, so that we can write, for all $p$ and $\bar{p}$

$$\Pi(\bar{p}) \leq \Pi(p) + \Pi'(p)(\bar{p} - p) + \frac{L}{2}(\bar{p} - p)^2$$

(This is closely related to a concept called Lipschitz continuity, beyond the scope of this course.)

# Proof of convergence (Optional)

Now, let's try $p = p_t$ and $\bar{p} = p_{t+1}$. We get

$$\Pi(p_{t+1}) \leq \Pi(p_t) + \Pi'(p_t)(p_{t+1} - p_t) + \frac{L}{2}(p_{t+1} - p_t)$$

Using $p_{t+1} = p_t - \sigma_t \Pi'(p_t)$, we find

$$\Pi(p_{t+1}) \leq \Pi(p_t) - \left(1 - \frac{L\sigma_t}{2}\right) \sigma_t \left[\Pi'(p_t)\right]^2$$

# Proof of convergence (Optional)

We found that

$$\Pi(p_{t+1}) \leq \Pi(p_t) - \left(1 - \frac{L\sigma_t}{2}\right)\sigma_t \left[\Pi'(p_t)\right]^2$$

this proves that our gradient descent converges for any $\sigma_t < 2/L$.

# Simple gradient descent – some issues

- What is $L$?

# Simple gradient descent – some issues

- What is $L$?
- Local minima/maxima

# Simple gradient descent – some issues

- What is $L$?
- Local minima/maxima
- Flat functions

# Simple gradient descent – some issues

- What is $L$?
- Local minima/maxima
- Flat functions
- Slow convergence

# Simple gradient descent – some issues

- What is $L$?
- Local minima/maxima
- Flat functions
- Slow convergence
- Uncertain functions

# Outline

# Global, gradient-free optimzation

We will now consider an optimization method that

- Doesn't require the gradient/slope.

# Global, gradient-free optimzation

We will now consider an optimization method that

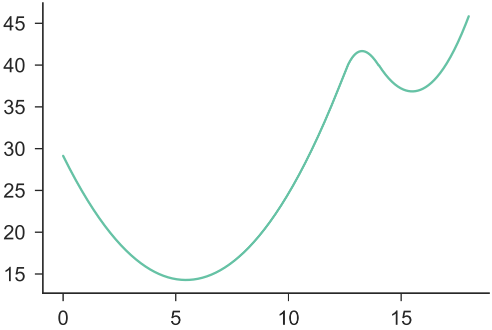
- Doesn't require the gradient/slope.
- Explores local minima.

# Global, gradient-free optimzation

We will now consider an optimization method that

- Doesn't require the gradient/slope.
- Explores local minima.
- Does not require the knowledge of any parameters relating to the function being optimized (like $L$ above).
- Easily extends to multiple dimensions, in the case of inter-dependent demand across products.

# The function to be minimized




The idea: subdivide the range of this function repeatedly to focus in on areas of interest.

# The Lipschitz Constant

We will assume that there exists a positive constant $L$ (called the *Lipschitz constant* such that

$$|\Pi(p) - \Pi(\bar{p})| \leq L|p - \bar{p}|$$

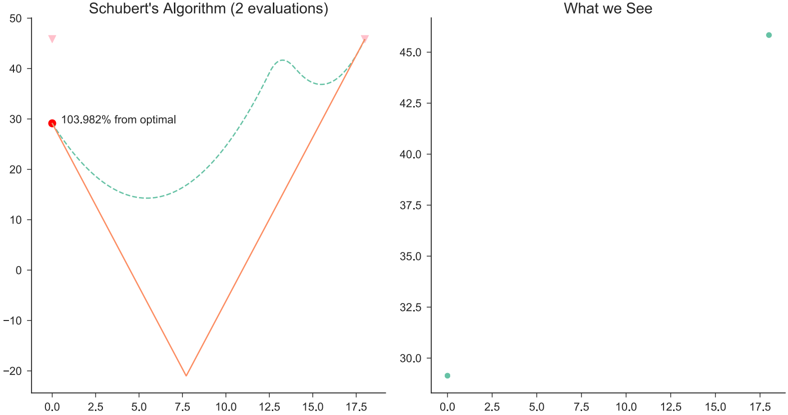
for any $p$ and $\bar{p}$.

(This is, once again, closely related to a concept called Lipschitz continuity, beyond the scope of this course.)

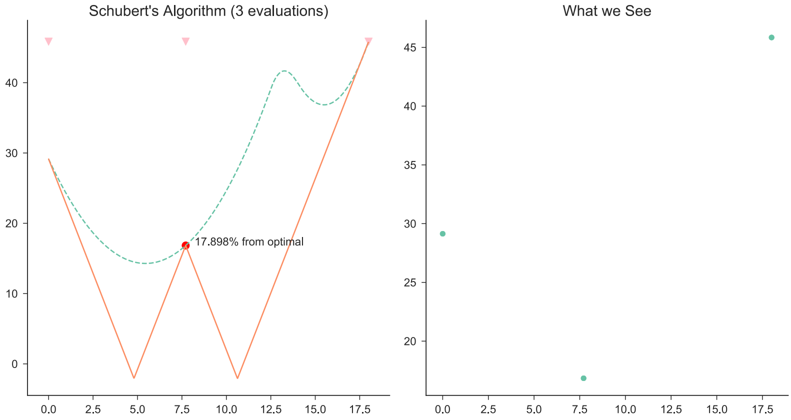
# Schubert's Algorithm

- Build the Lipschitz Lower Bound
- Identify the most promising interval by finding the lowest part of the lower bound
- Sub-divide that interval at its lower point
- Repeat

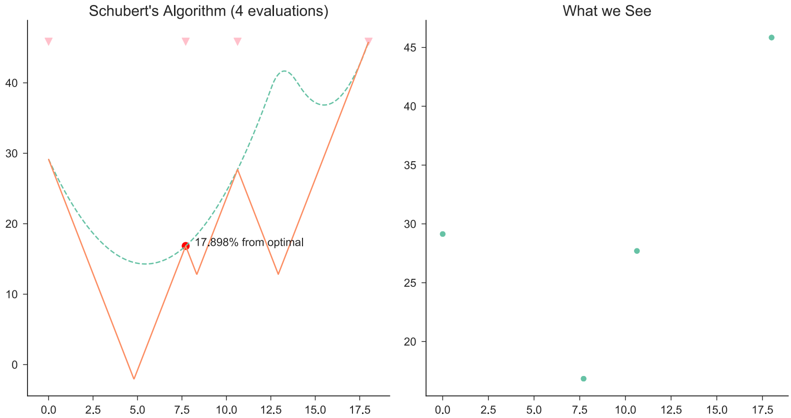
# Schubert's Algorithm



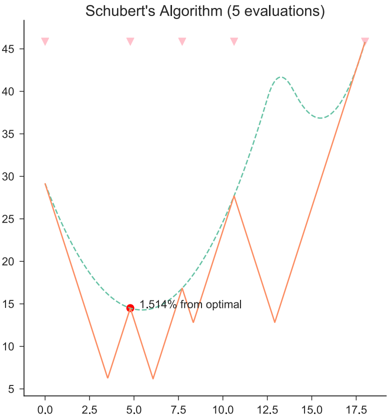
# Schubert's Algorithm

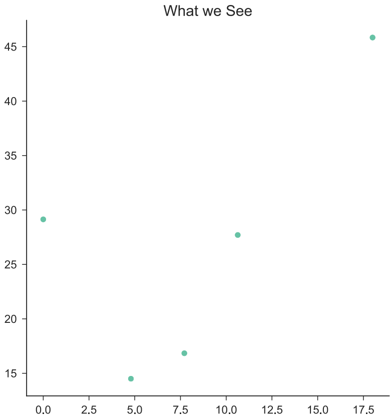


# Schubert's Algorithm

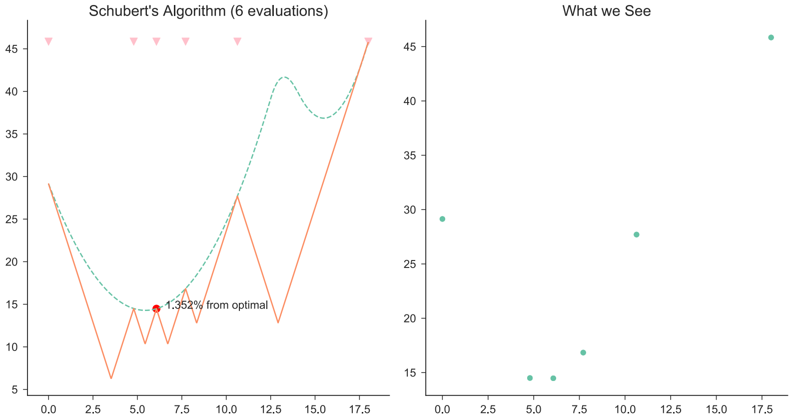


# Schubert's Algorithm

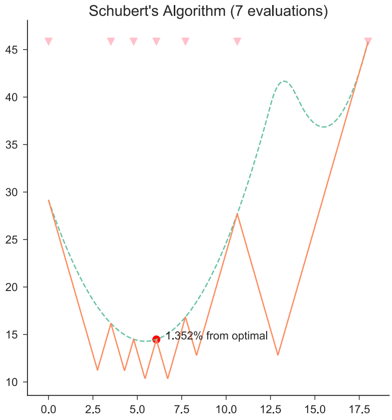


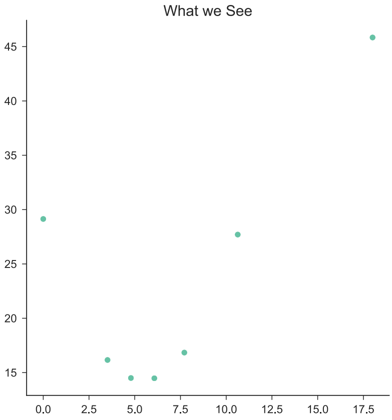



# Schubert's Algorithm

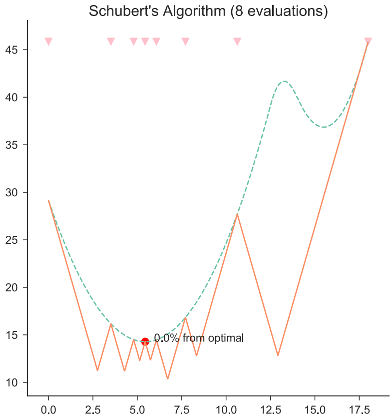


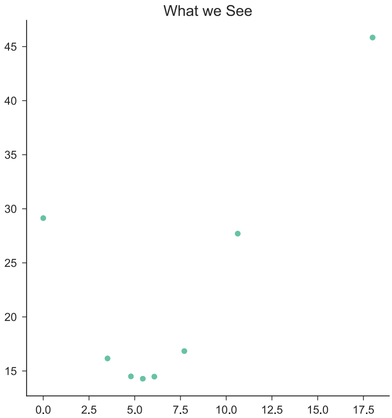
# Schubert's Algorithm



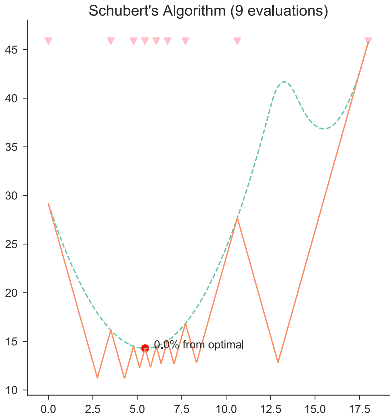



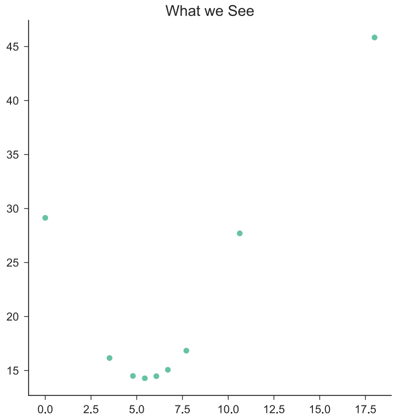
# Schubert's Algorithm



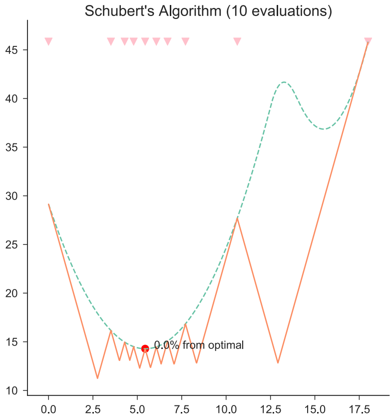



# Schubert's Algorithm

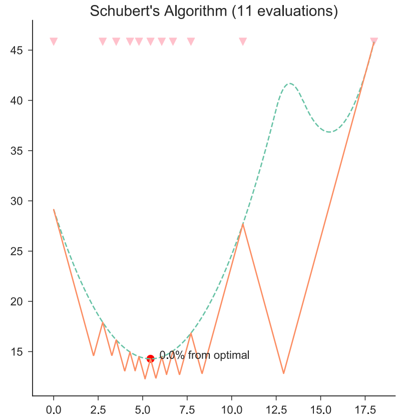


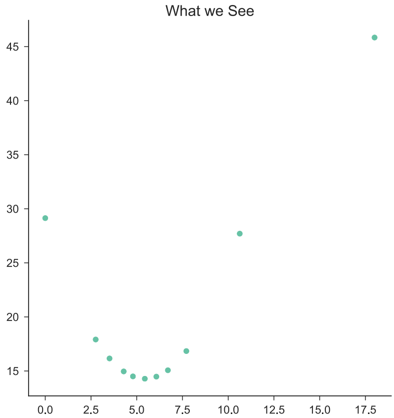



# Schubert's Algorithm

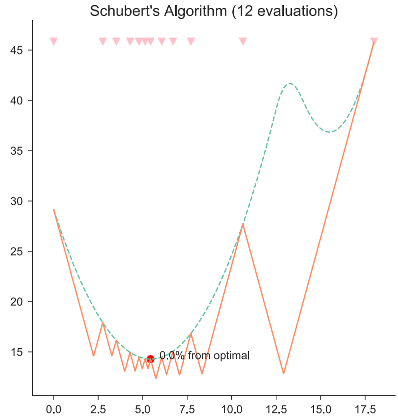


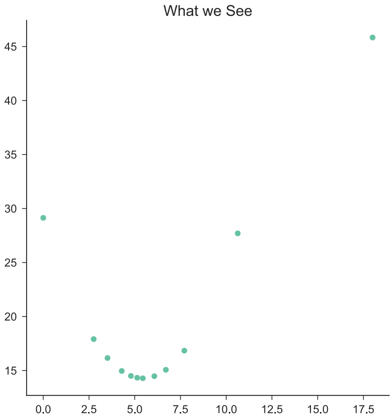
# Schubert's Algorithm



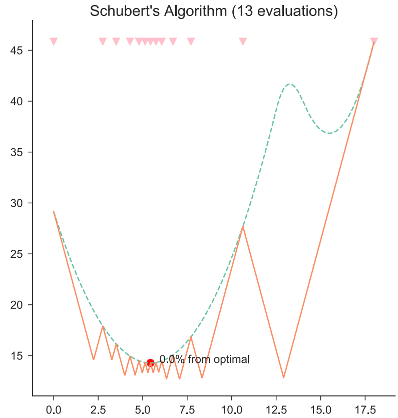



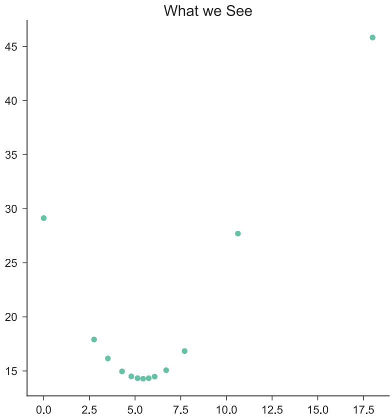
# Schubert's Algorithm



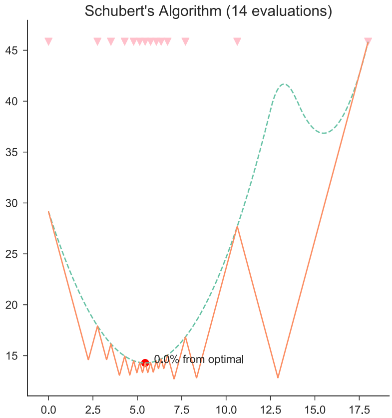



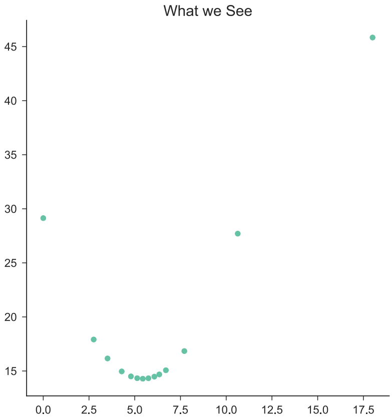
# Schubert's Algorithm



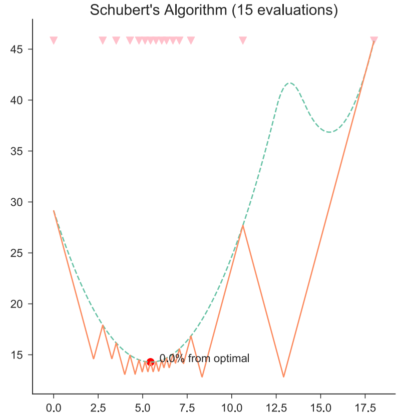



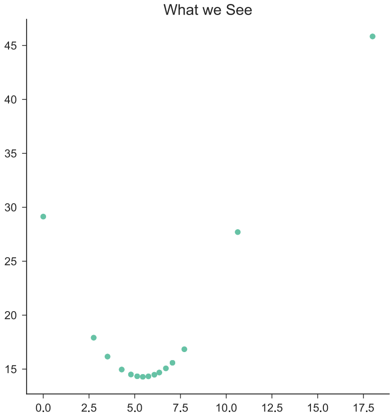
# Schubert's Algorithm



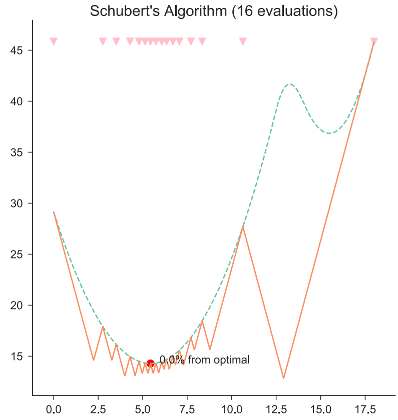



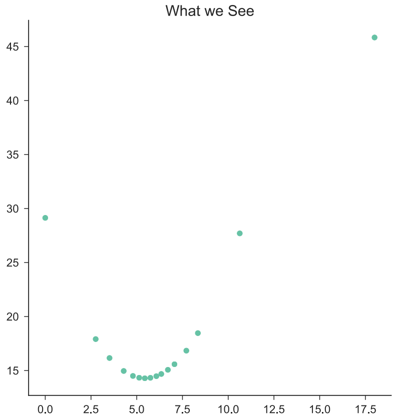
# Schubert's Algorithm



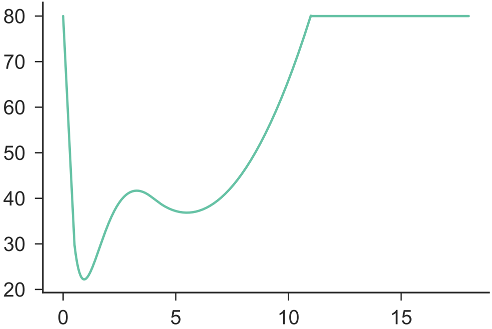



# Schubert's Algorithm

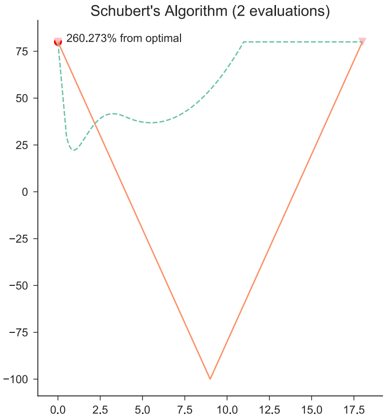


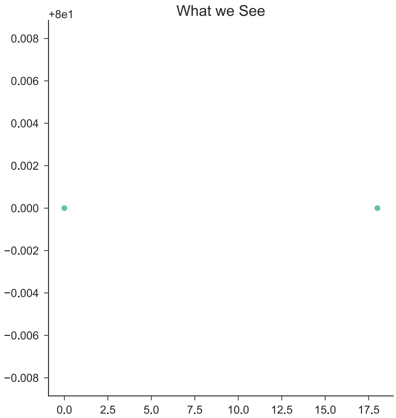



# Picking the wrong $L$

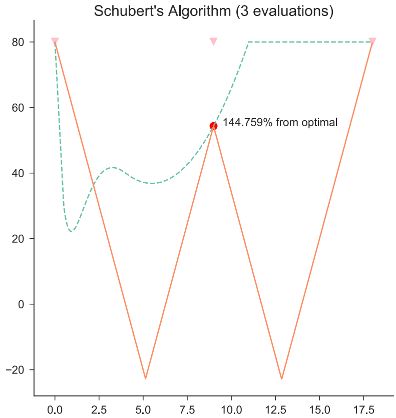


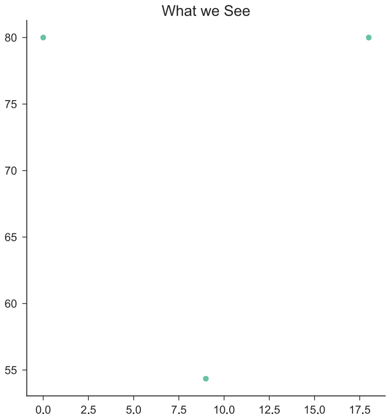
# Picking the wrong $L$



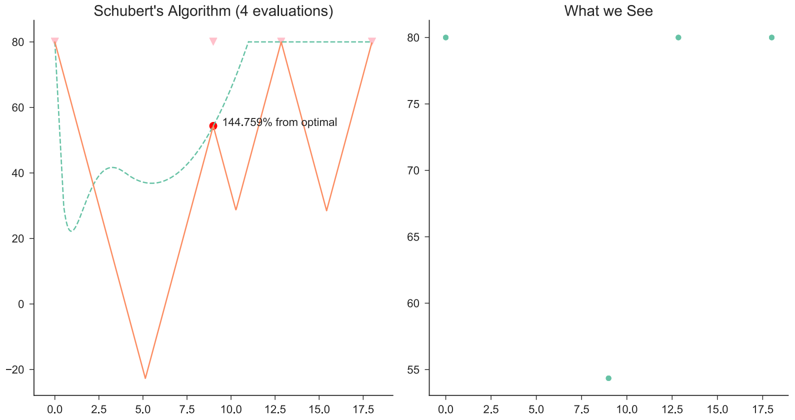



# Picking the wrong $L$

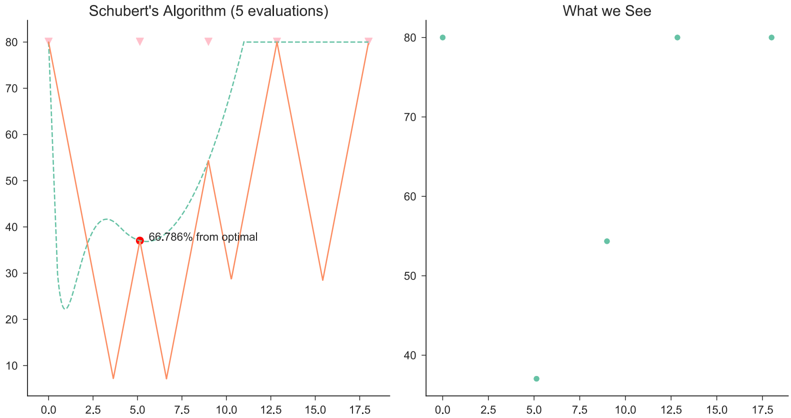


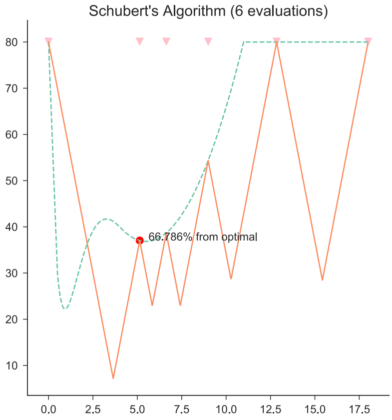
# Picking the wrong $L$

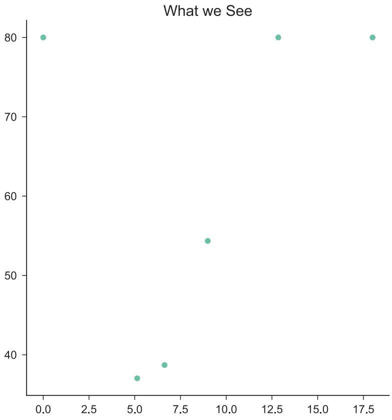


# Picking the wrong $L$

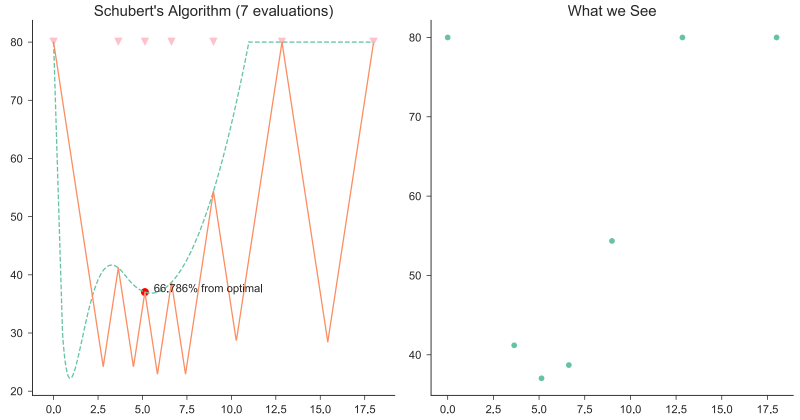


# Picking the wrong $L$

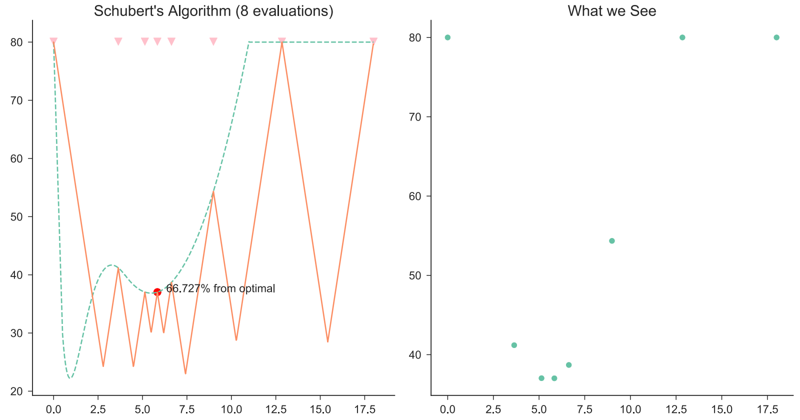


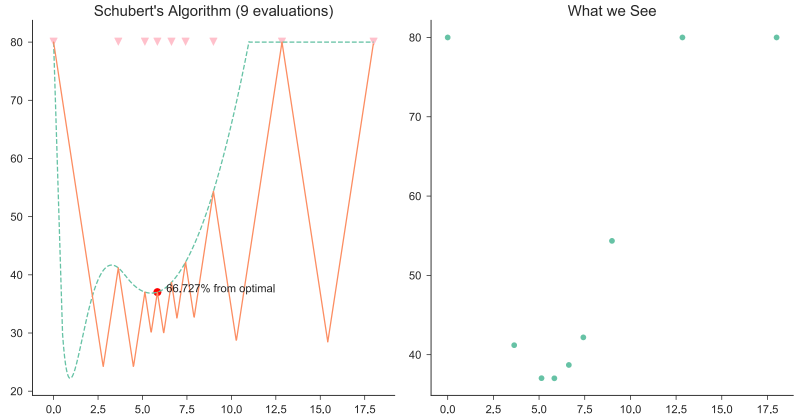
# Picking the wrong $L$



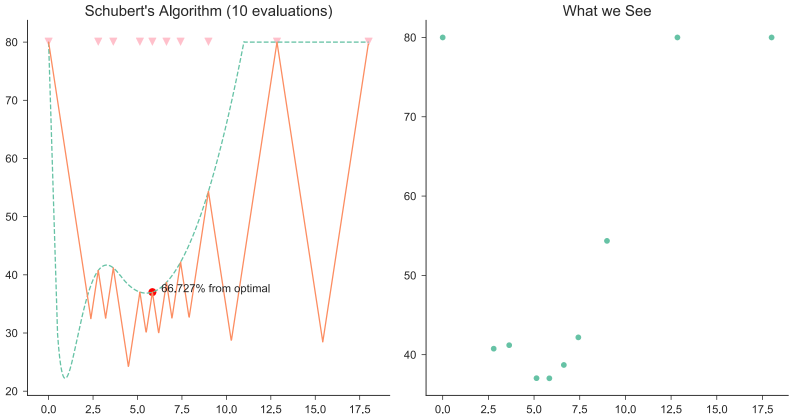
# Picking the wrong $L$



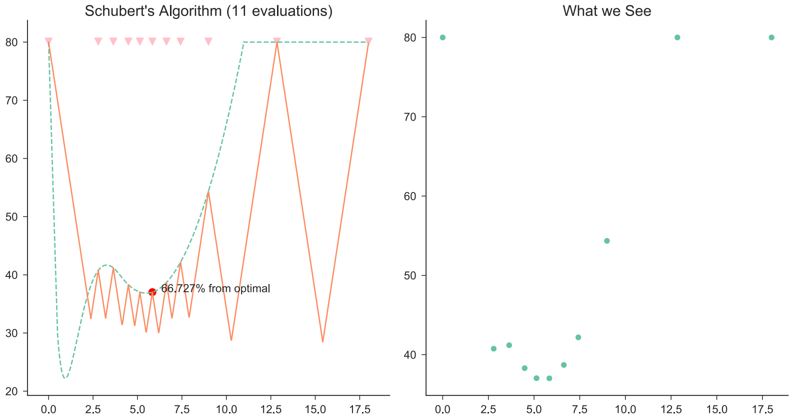
# Picking the wrong $L$



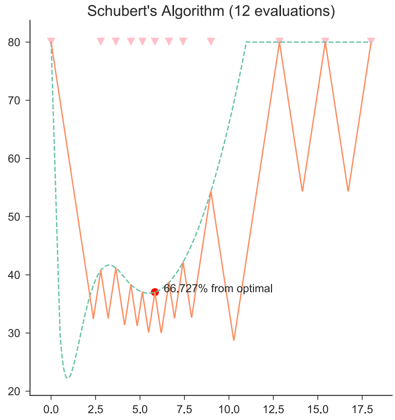
# Picking the wrong $L$

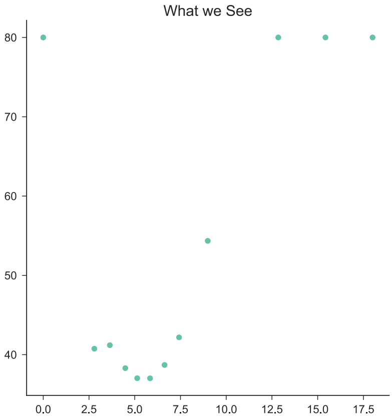


# Picking the wrong $L$

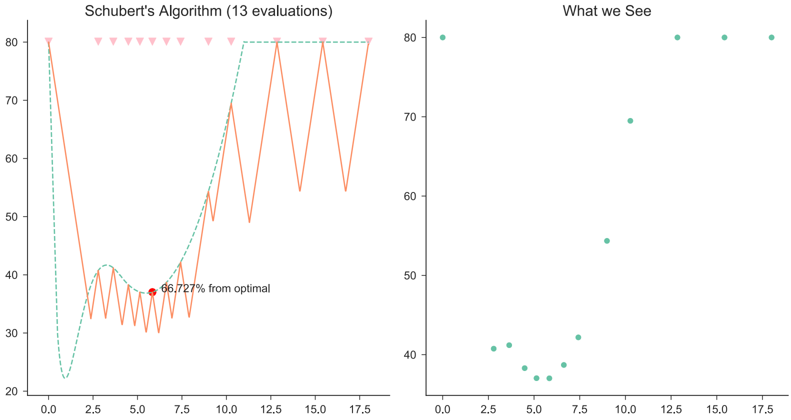


# Picking the wrong $L$

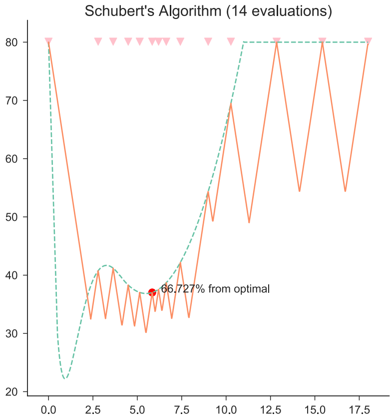


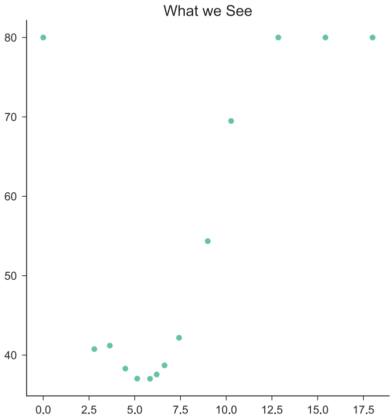



# Picking the wrong $L$

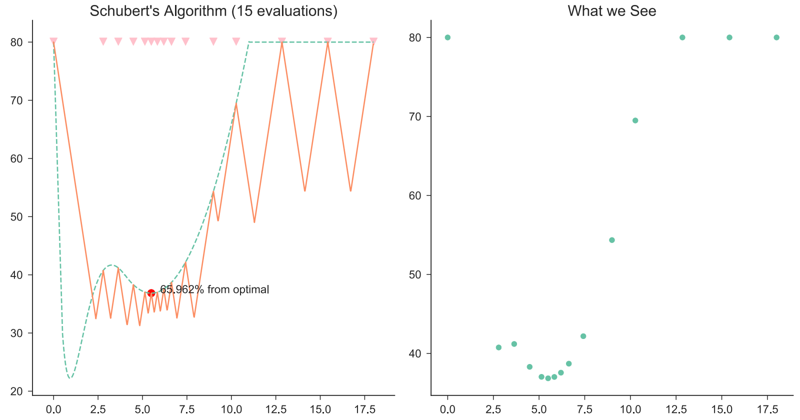


# Picking the wrong $L$

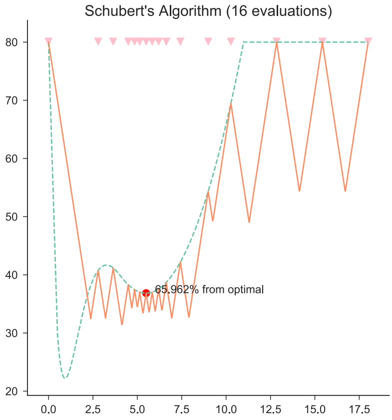


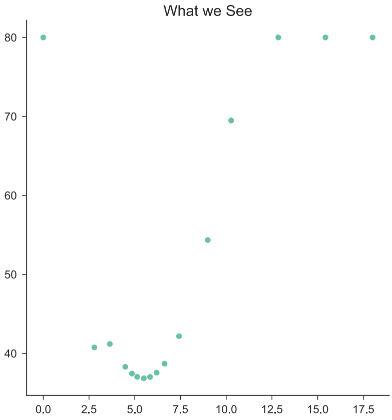



# Picking the wrong $L$



# Picking the wrong $L$



# A further look at the role of $L$

Suppose we have split the range of our function into an number of intervals $[p_i, p_j]$. The next step in Schubert's Algorithm will choose to further split the interval with the lowest Lipschitz lower bound. What is an expression for this lower bound?
What does this imply about the role of $L$?

# Schubert's algorithm – some issues

- What is $L$?
- ~~Local minima/maxima~~
- ~~Flat functions~~
- Slow convergence
- ~~Uncertain functions~~
- What about higher dimensions?

# The DIRECT Algorithm

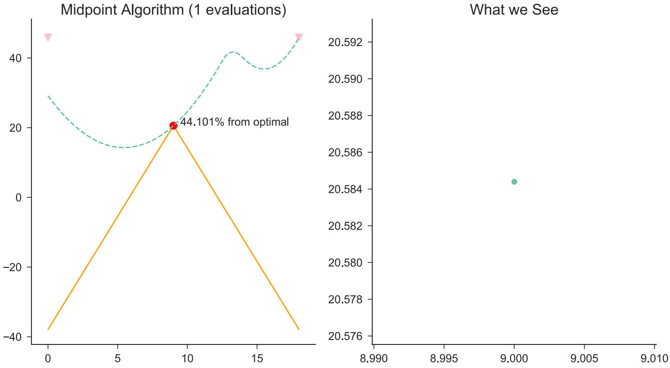
Two main changes

- Sample at the *center* of each interval rather than at its edges.
- Try every possible value of $L$ at every step.

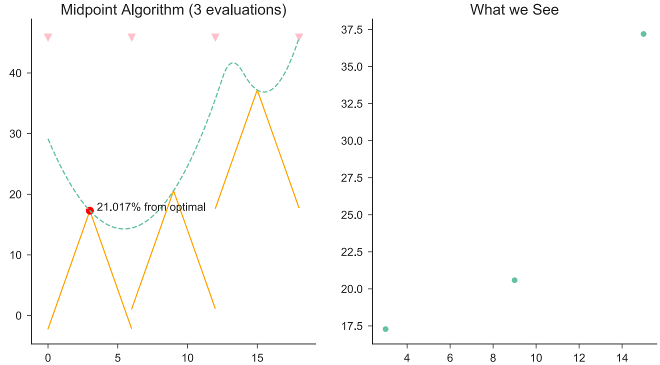
# Sampling at the *center* of each interval

What does sampling at the center of each interval look like?

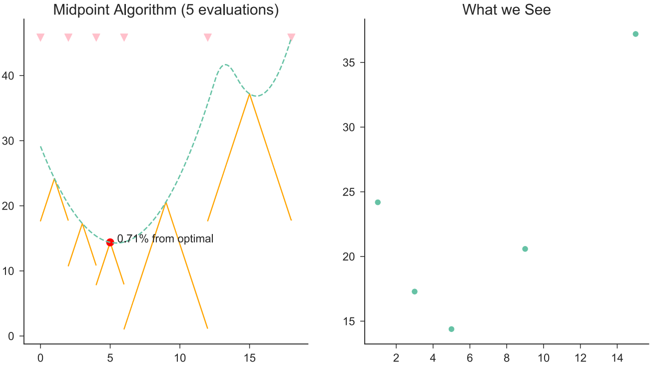
# Sampling at the *center* of each interval



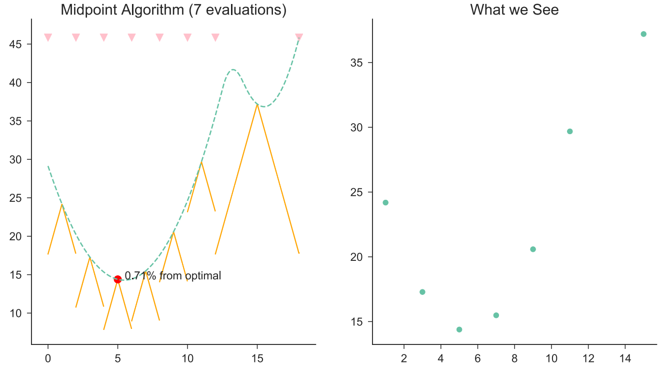
# Sampling at the *center* of each interval



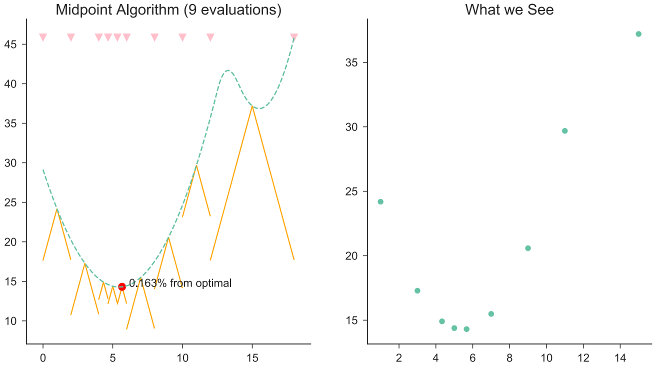
# Sampling at the *center* of each interval



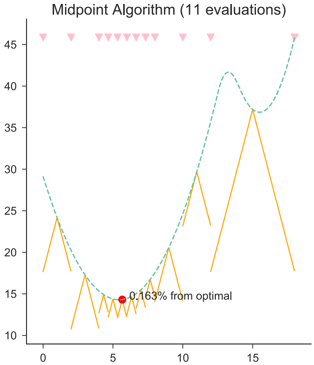
# Sampling at the *center* of each interval

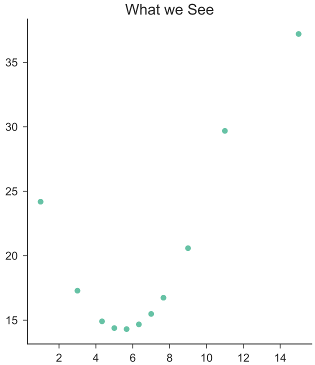


# Sampling at the *center* of each interval

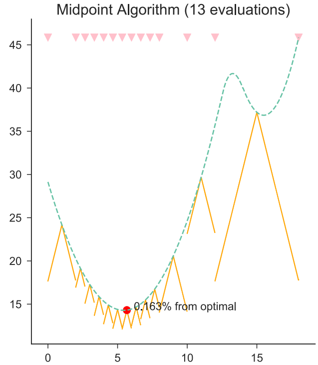


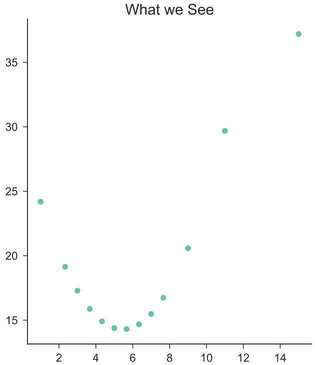
# Sampling at the *center* of each interval



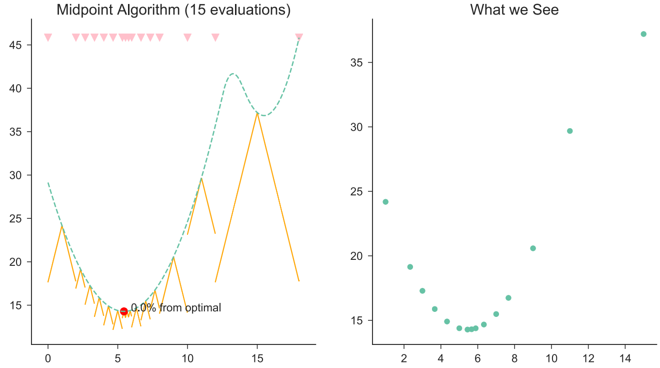



# Sampling at the *center* of each interval

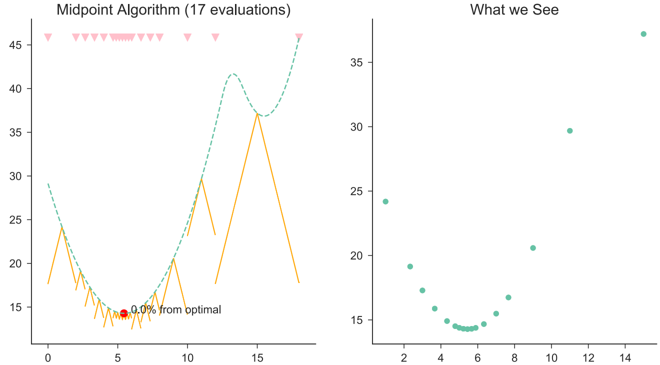


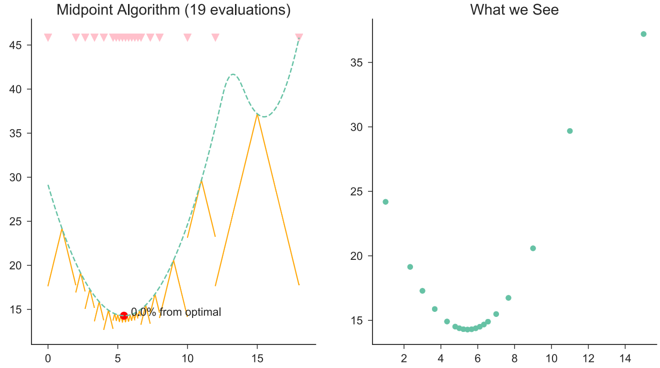
# Sampling at the *center* of each interval



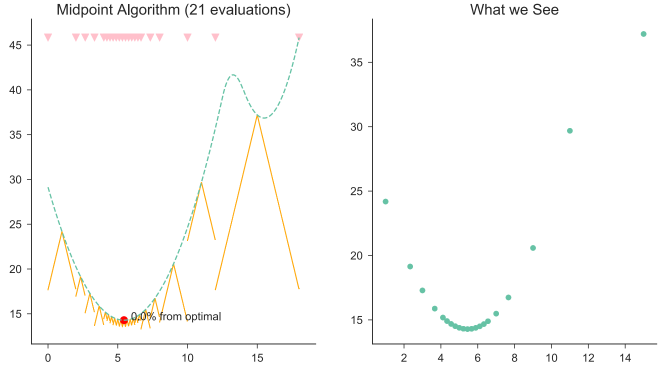
# Sampling at the *center* of each interval



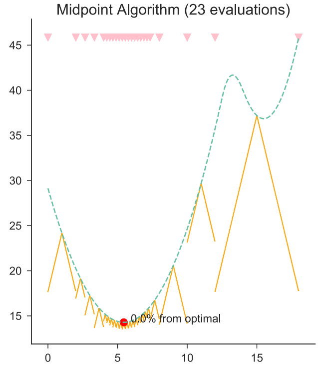
# Sampling at the *center* of each interval

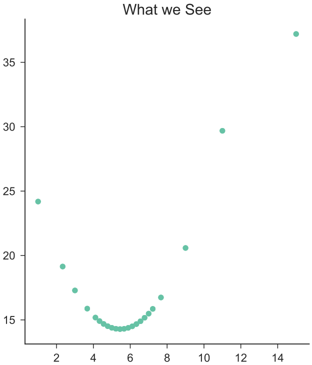


# Sampling at the *center* of each interval

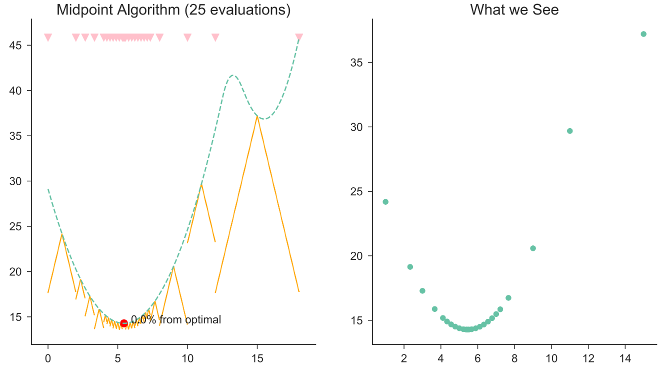


# Sampling at the *center* of each interval

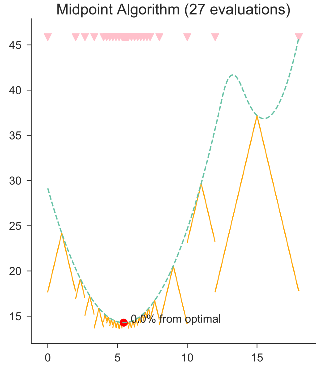


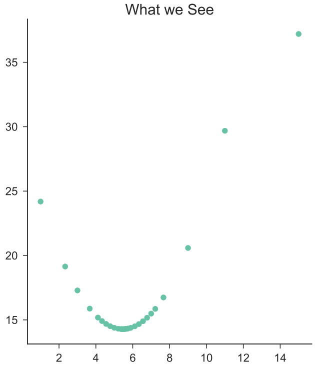



# Sampling at the *center* of each interval

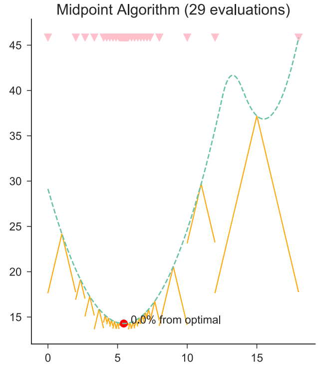


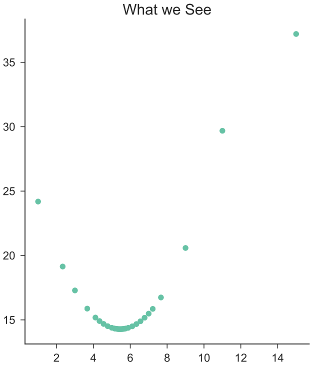
# Sampling at the *center* of each interval



# Sampling at the *center* of each interval

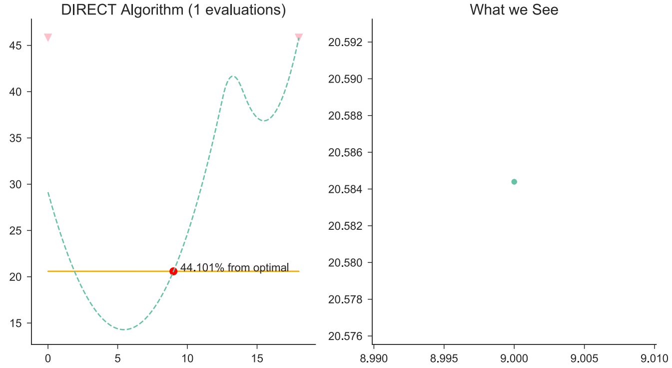


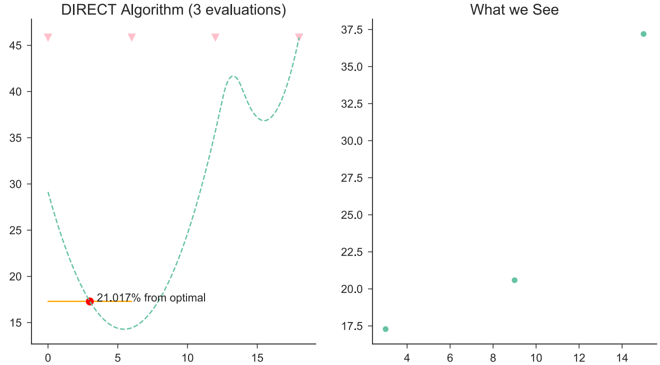
# Trying every possible value of $L$ at every step

How would we do this?

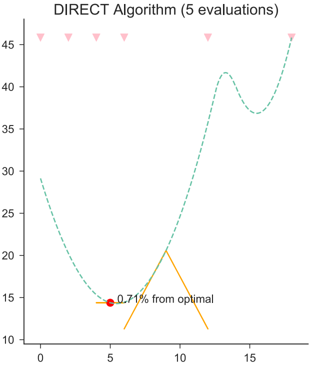
# Trying every possible value of $L$ at every step

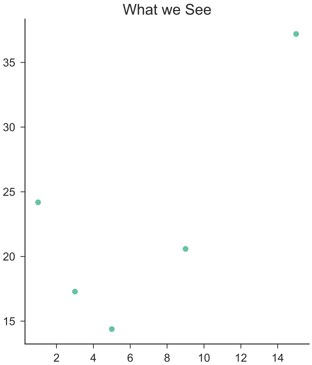


# Trying every possible value of $L$ at every step

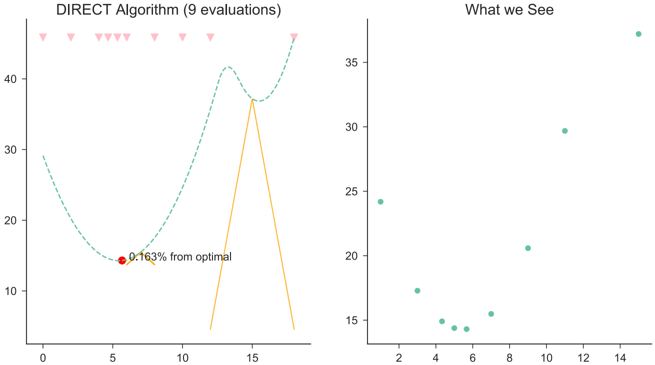


# Trying every possible value of $L$ at every step

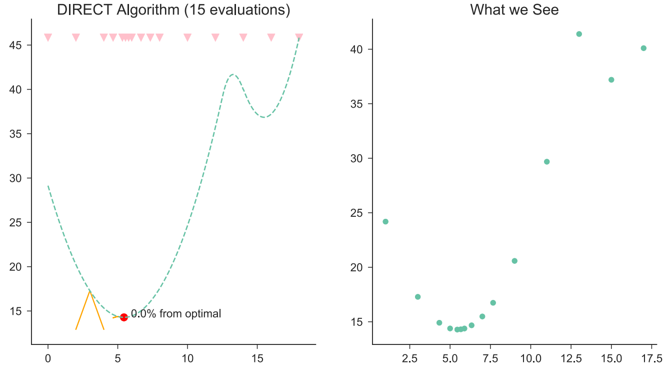


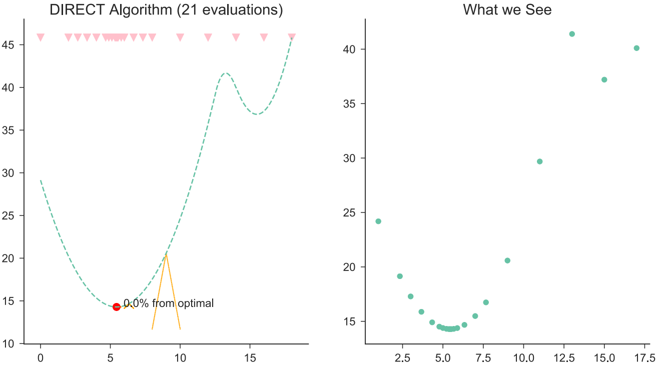
# Trying every possible value of $L$ at every step



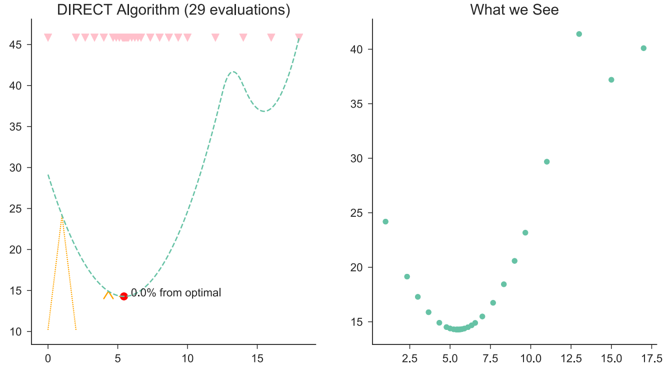
# Trying every possible value of $L$ at every step



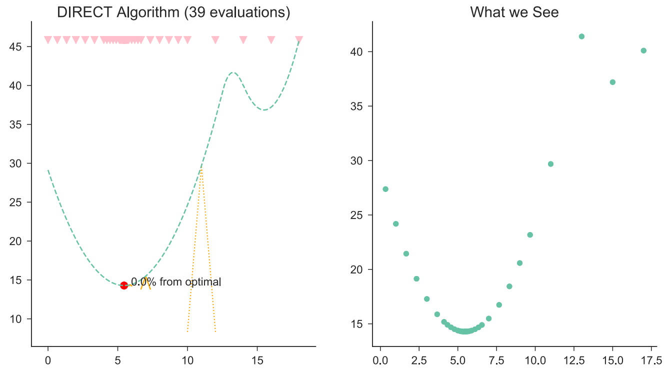
# Trying every possible value of $L$ at every step



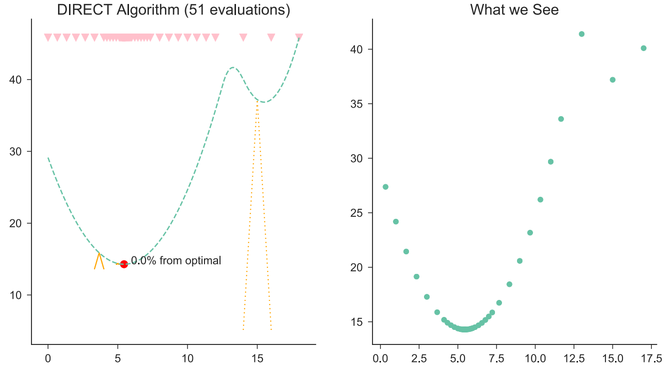
# Trying every possible value of $L$ at every step



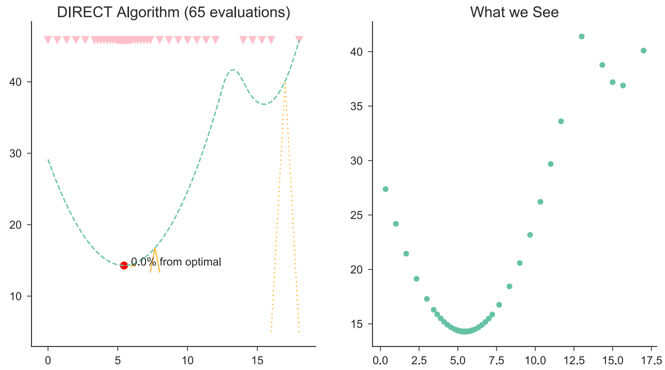
# Trying every possible value of $L$ at every step



# Trying every possible value of $L$ at every step



# Trying every possible value of $L$ at every step



# Trying every possible value of $L$ at every step