STAN WORKSHOP

A PROBABILITY MODEL FOR GOLF PUTTING

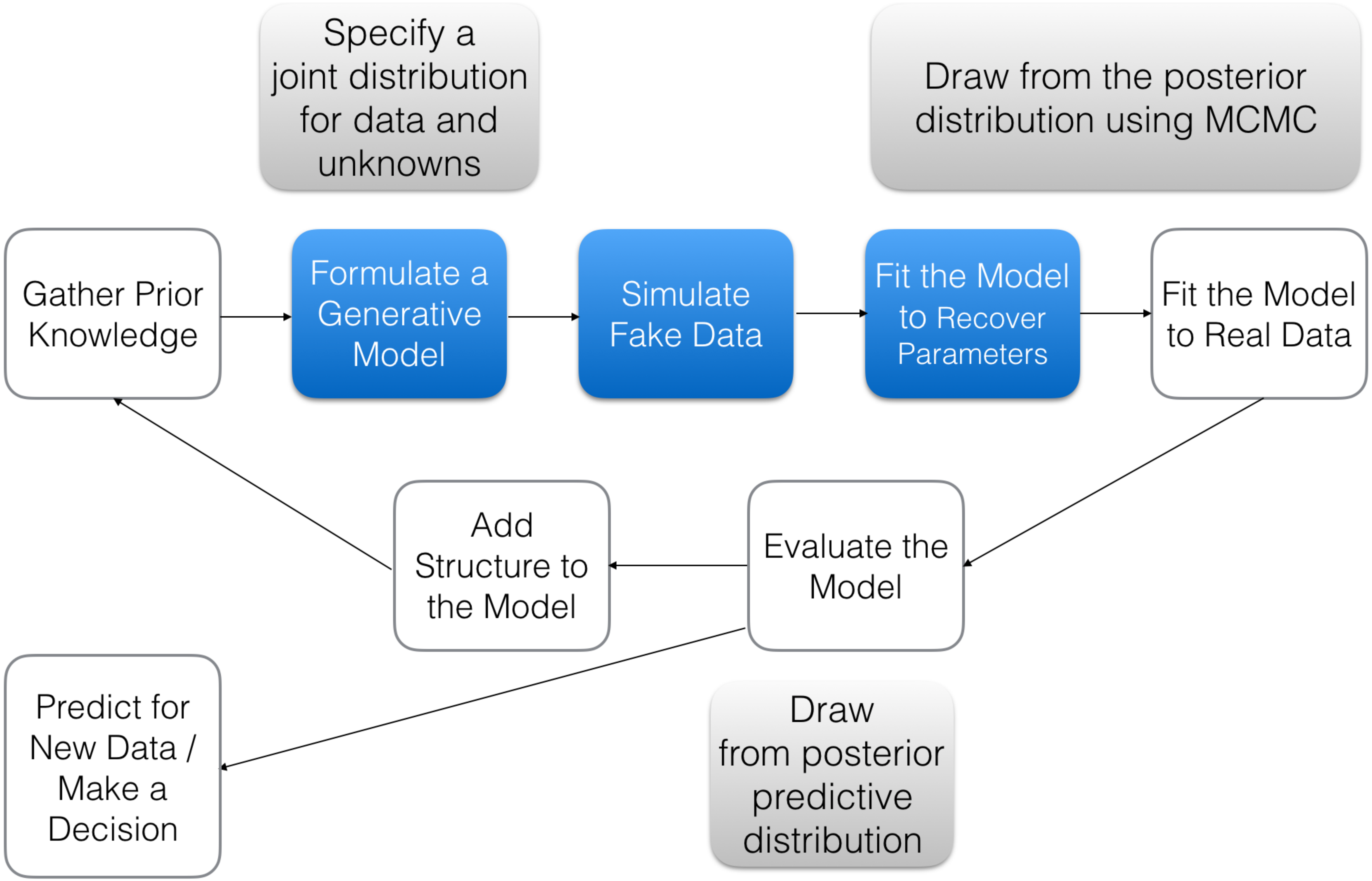
INTRO

THANK YOU

- Google: Sonia, Joesph, + others
- http://www.meetup.com/bda-group/
- Stan Group: http://stan.fit
- Daniel
bearlee@alum.mit.edu

  Eric
eric@stan.fit

BAYESIAN WORKFLOW



WHAT IS STAN?

LANGUAGE, ALGORITHMS, INTERFACES

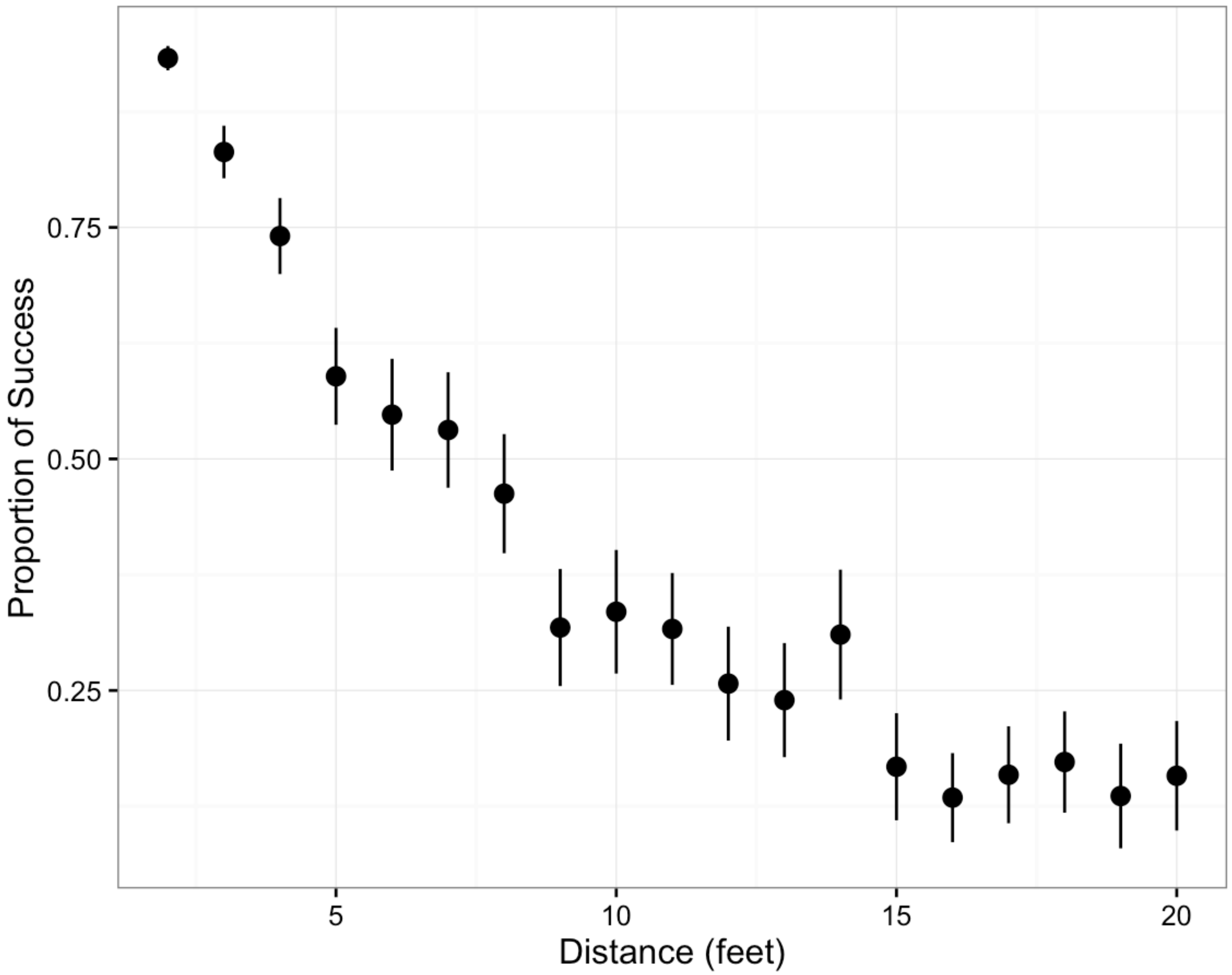
- Language: specify statistical models
  - Define data y, X
  - Define parameters θ
  - Define statistical model

$$
\begin{aligned}
&\log p(y, X, \theta) \\
&\quad = \log \Big(p(\theta) \times p(y, X \mid \theta) \Big) \\
&\quad = \log p(\theta) + \log p(y, X \mid \theta)
\end{aligned}
$$

LANGUAGE, ALGORITHMS, INTERFACES

```
> plot(successes / tries, ylim = c(0, 1))
```

- Define data (type this in R):
```
N <- 19
tries <- c(1443, 694, 455, 353,
  272, 256, 240, 217, 200, 237,
  202, 192, 174, 167, 201, 195,
  191, 147, 152)
successes <- c(1346, 577, 337,
  208, 149, 136, 111, 69, 67, 75,
  52, 46, 54, 28, 27, 31, 33, 20,
  24)
```
- Define parameters
  probability of success: `p[N]`
- Define statistical model
  every p uniform in [0, 1]
  independent Binomial distributions



LANGUAGE, ALGORITHMS, INTERFACES

- Define data (type this in R):

```
N <- 19
tries <- c(1443, 694, 455, 353,
  272, 256, 240, 217, 200, 237,
  202, 192, 174, 167, 201, 195,
  191, 147, 152)
successes <- c(1346, 577, 337,
  208, 149, 136, 111, 69, 67, 75,
  52, 46, 54, 28, 27, 31, 33, 20,
  24)
```

- Define parameters
  probability of success: p[N]
- Define statistical model
  every p uniform in [0, 1]
  independent Binomial distributions

example.stan

```
data {
  int N;
  int<lower = 0> tries[N];
  int<lower = 0> successes[N];
}
```

LANGUAGE, ALGORITHMS, INTERFACES

- Define data (type this in R):
```
N <- 19
tries <- c(1443, 694, 455, 353,
  272, 256, 240, 217, 200, 237,
  202, 192, 174, 167, 201, 195,
  191, 147, 152)
successes <- c(1346, 577, 337,
  208, 149, 136, 111, 69, 67, 75,
  52, 46, 54, 28, 27, 31, 33, 20,
  24)
```
- Define parameters
  probability of success: p[N]
- Define statistical model
  every p uniform in [0, 1]
  independent Binomial distributions

example.stan

```
data {
  int N;
  int<lower = 0> tries[N];
  int<lower = 0> successes[N];
}

parameters {
  real<lower = 0, upper = 1> p[N];
}
```

LANGUAGE, ALGORITHMS, INTERFACES

- Define data (type this in R):

```
N <- 19
tries <- c(1443, 694, 455, 353,
  272, 256, 240, 217, 200, 237,
  202, 192, 174, 167, 201, 195,
  191, 147, 152)
successes <- c(1346, 577, 337,
  208, 149, 136, 111, 69, 67, 75,
  52, 46, 54, 28, 27, 31, 33, 20,
  24)
```

- Define parameters
  probability of success: p[N]
- Define statistical model
  every p uniform in [0, 1]
  independent Binomial distributions

example.stan

```
data {
  int N;
  int<lower = 0> tries[N];
  int<lower = 0> successes[N];
}

parameters {
  real<lower = 0, upper = 1> p[N];
}

model {
  p ~ uniform(0, 1);

  for (n in 1:N) {
    successes[n] ~ binomial(tries[n], p[n]);
  }

}
```

LANGUAGE, ALGORITHMS, INTERFACES

- Define data (type this in R):
  ```
  N <- 19
  tries <- c(1443, 694, 455, 353,
    272, 256, 240, 217, 200, 237,
    202, 192, 174, 167, 201, 195,
    191, 147, 152)
  successes <- c(1346, 577, 337,
    208, 149, 136, 111, 69, 67, 75,
    52, 46, 54, 28, 27, 31, 33, 20,
    24)
  ```
- Define parameters
  probability of success: p[N]
- Define statistical model
  every p uniform in [0, 1]
  independent Binomial distributions

example.stan

```
data {
  int N;
  int<lower = 0> tries[N];
  int<lower = 0> successes[N];
}

parameters {
  real<lower = 0, upper = 1> p[N];
}

model {
  p ~ uniform(0, 1);

  for (n in 1:N) {
    successes[n] ~ binomial(tries[n], p[n]);
  }

}
```

LANGUAGE, ALGORITHMS, INTERFACES

- Define data (type this in R):

```
N <- 19
tries <- c(1443, 694, 455, 353,
  272, 256, 240, 217, 200, 237,
  202, 192, 174, 167, 201, 195,
  191, 147, 152)
successes <- c(1346, 577, 337,
  208, 149, 136, 111, 69, 67, 75,
  52, 46, 54, 28, 27, 31, 33, 20,
  24)
```

- Define parameters
  probability of success: p[N]
- Define statistical model
  every p uniform in [0, 1]
  independent Binomial distributions

example.stan

```
data {
  int N;
  int<lower = 0> tries[N];
  int<lower = 0> successes[N];
}

parameters {
  real<lower = 0, upper = 1> p[N];
}

model {
  successes ~ binomial(tries, p);
}
```

example.R

```
> fit <- stan("example.stan",
              data = list(N, tries, successes))
```

LANGUAGE, ALGORITHMS, INTERFACES

- Inference algorithms
  - Full Markov Chain Monte Carlo (MCMC)
    - **No-U-Turn Sampler (NUTS)**
    - HMC
  - Approximate Bayesian inference: ADVI
  - Optimization: L-BFGS, BFGS, Newton

LANGUAGE, ALGORITHMS, INTERFACES

- Tight integration
  - CmdStan
  - PyStan
  - **RStan**
    - shinystan
    - loo
    - rstanarm
- Process level
  - MatlabStan
  - Stan.jl
  - StataStan

PUTTING

DATA: PGA

```
N <- 19

tries <- c(1443, 694, 455, 353,
  272, 256, 240, 217, 200, 237,
  202, 192, 174, 167, 201, 195,
  191, 147, 152)

successes <- c(1346, 577, 337,
  208, 149, 136, 111, 69, 67, 75,
  52, 46, 54, 28, 27, 31, 33, 20,
  24)

distance <- 2:20
```

DATA: PGA

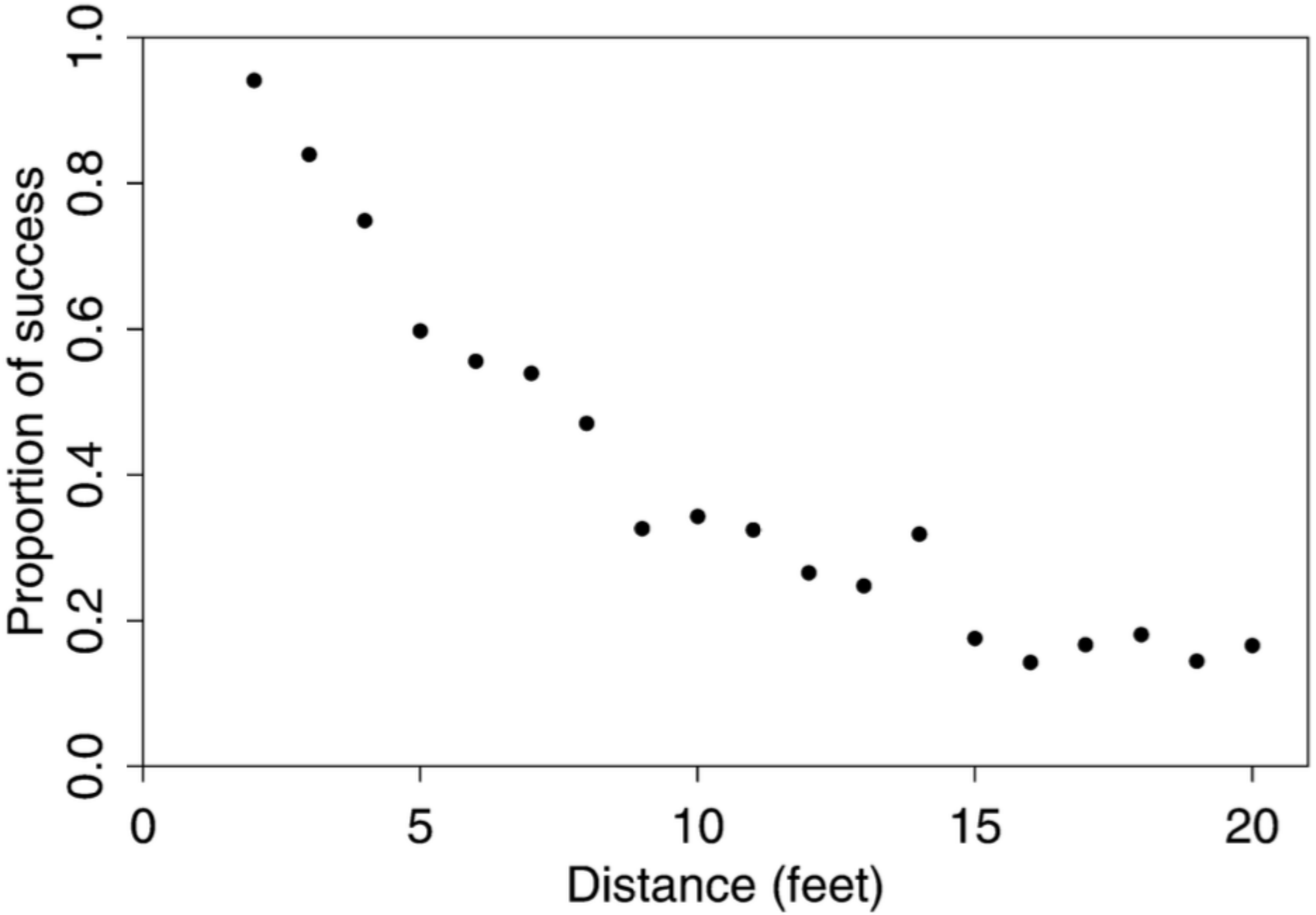
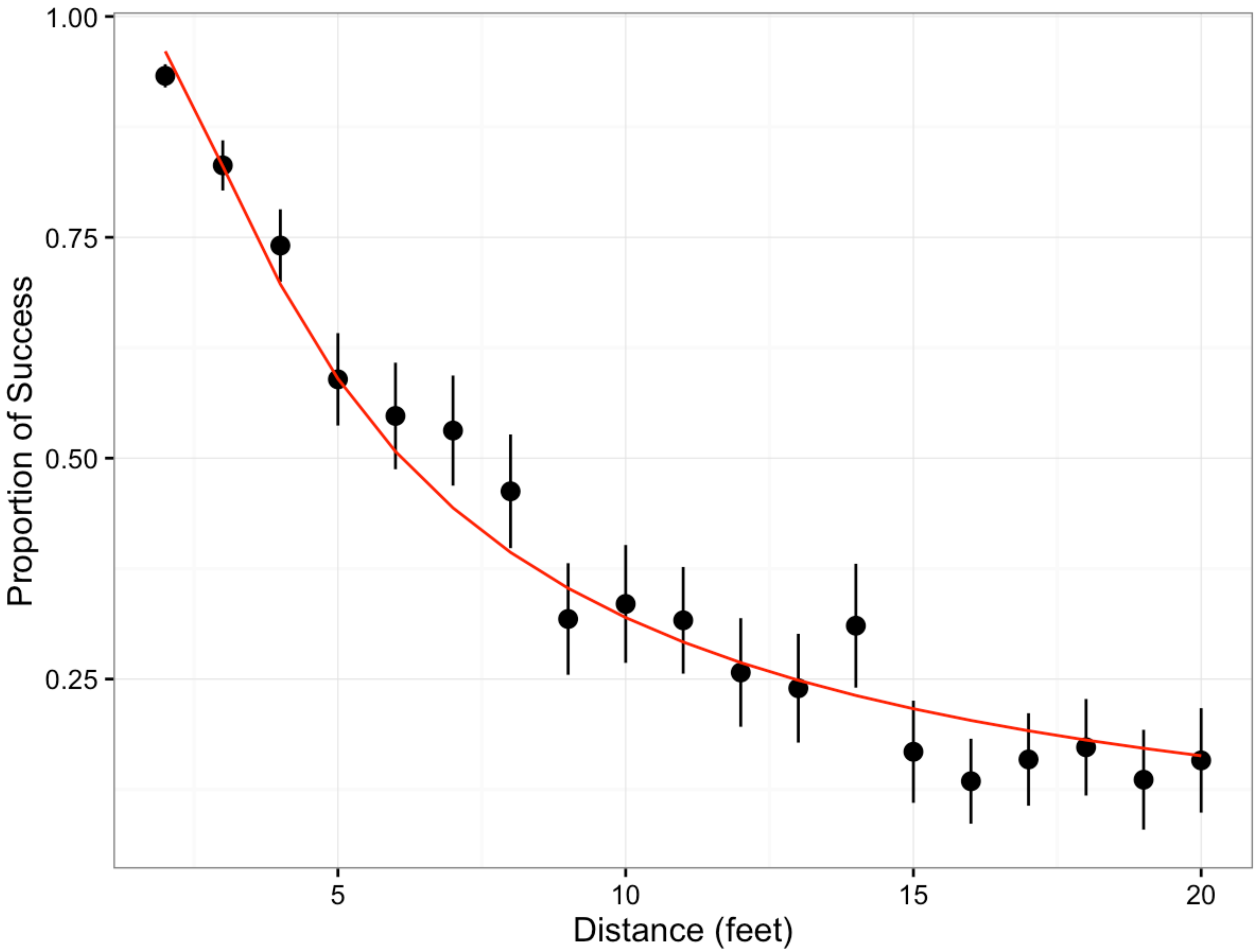
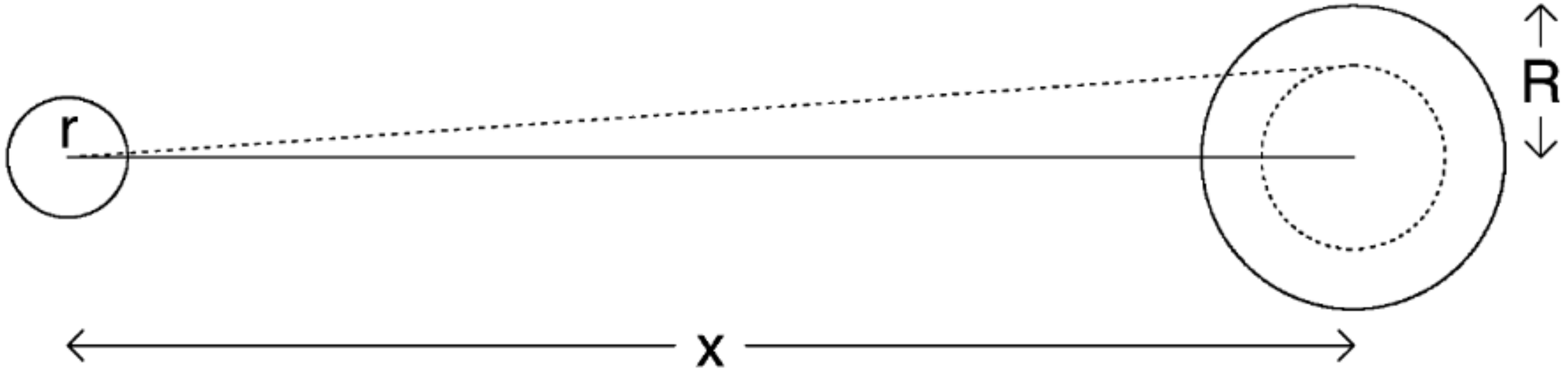


Fig 1. Success rate of golf putts as a function of distance from the hole

FIT A CURVE USING THE STATED PARAMETER FROM THE PAPER (SIGMA = 0.026 RAD)

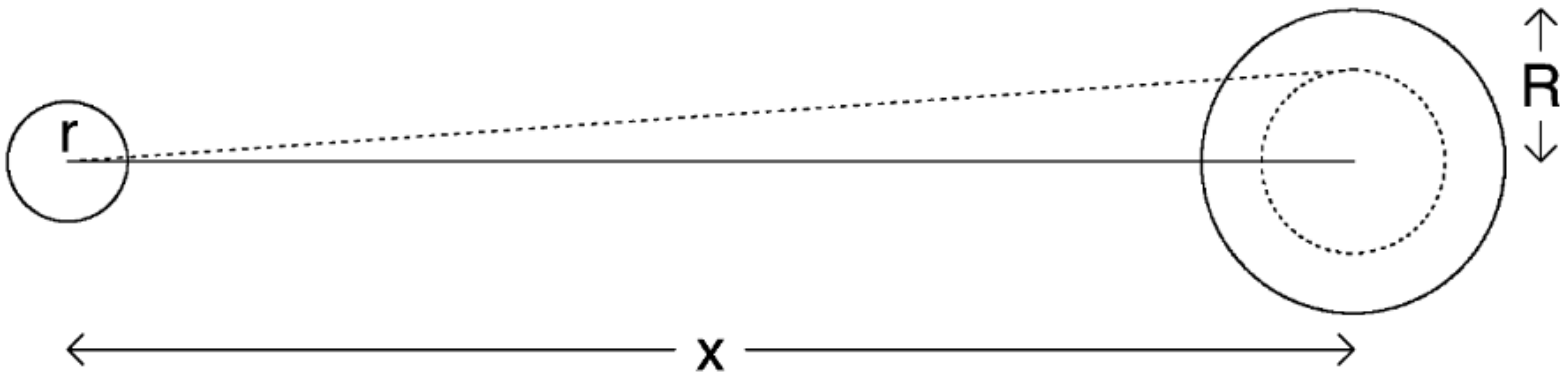
```
golf <- mutate(golf,
           p = successes / tries,
           error_sd = sqrt((p * (1 - p))
                               / tries),
           lower = p - 2 * error_sd,
           upper = p + 2 * error_sd,
           fit = 2 * pnorm(theta0(dist * 12) /
                                sigma) - 1)

limits <- with(golf, aes(ymax = upper,
                         ymin = lower))
p <- ggplot(golf, aes(x = dist, y = p))
p <- p + geom_pointrange(limits)
p <- p + geom_line(aes(y = fit),
                     colour = "red")
p +  xlab("Distance (feet)") +
     ylab("Proportion of Success")

theta0 <- function(x) {
  asin((R - r) / x)
}
```





2r = 1.68 inches

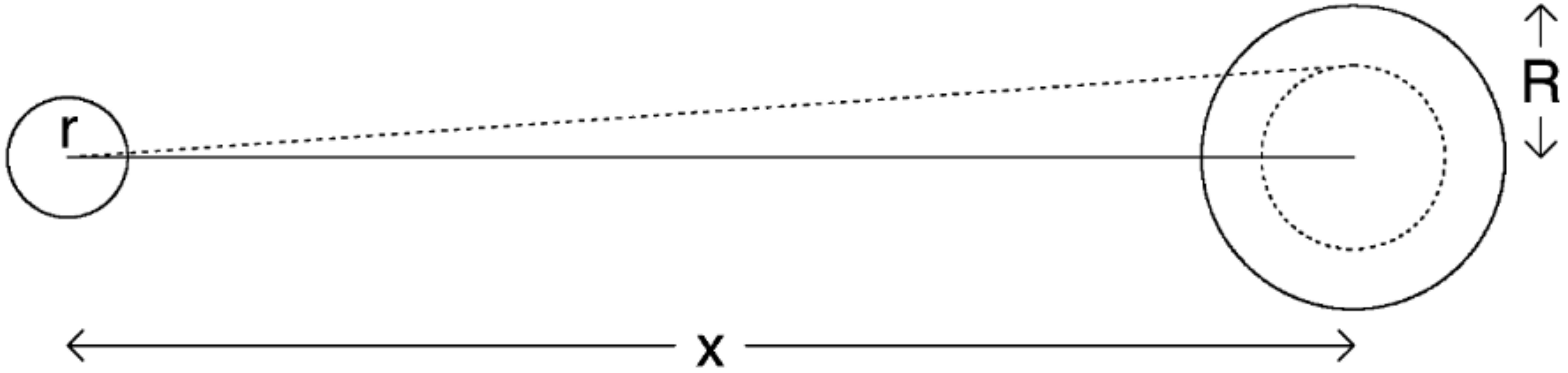
2R = 4.25 inches



$2r = 1.68$ inches

$2R = 4.25$ inches

THRESHOLD ANGLE:



2r = 1.68 inches

2R = 4.25 inches

THRESHOLD ANGLE:

$$\theta_0 = \arcsin\left(\frac{R-r}{x}\right)$$

$$P\,(\text{success of a shot from distance } x) = 2\Phi\left(\frac{1}{\sigma}\arcsin\left(\frac{R-r}{x}\right)\right) - 1$$

where Φ is the standard Normal cumulative distribution function. (If $x < R - r$, then $\arcsin\{(R-r)/x\}$ is not defined, but in this case the model is not needed since the ball is already in the hole!)

WHY IS THAT?

$$P\,(\text{success of a shot from distance } x) = 2\Phi\left(\frac{1}{\boxed{\sigma}}\arcsin\left(\frac{R-r}{x}\right)\right) - 1$$

parameter

Phi()

asin()

where Φ is the standard Normal cumulative distribution function. (If $x < R - r$, then $\arcsin\{(R-r)/x\}$ is not defined, but in this case the model is not needed since the ball is already in the hole!)

```
data {
  int N;
  int<lower = 0> tries[N];
  int<lower = 0> successes[N];
  real<lower = 0> dist[N];
}

parameters {

}

model {



}
```

$$
P\,(\text{success of a shot from distance } x) = 2\Phi\left(\frac{1}{\boxed{\sigma}}\arcsin\left(\frac{R-r}{x}\right)\right) - 1
$$

parameter

where Φ is the standard Normal cumulative distribution function. (If $x < R - r$, then $\arcsin\{(R - r)/x\}$ is not defined, but in this case the model is not needed since the ball is already in the hole!)

```
data {
  int N;
  int<lower = 0> tries[N];
  int<lower = 0> successes[N];
  real<lower = 0> dist[N];
}

parameters {
  real<lower = 0> sigma;
}

model {




}
```

$$P\,(\text{success of a shot from distance } x)$$

$$= 2\Phi\left(\frac{1}{\boxed{\sigma}}\arcsin\left(\frac{R-r}{x}\right)\right) - 1$$

parameter

Phi()

asin()

where Φ is the standard Normal cumulative distribution function. (If $x < R - r$, then $\arcsin\{(R-r)/x\}$ is not defined, but in this case the model is not needed since the ball is already in the hole!)

```
data {
  int N;
  int<lower = 0> tries[N];
  int<lower = 0> successes[N];
  real<lower = 0> dist[N];
}

parameters {
  real<lower = 0> sigma;
}

model {
  real p[N];









  successes ~ binomial(tries, p);
}
```

```
data {
  int N;
  int<lower = 0> tries[N];
  int<lower = 0> successes[N];
  real<lower = 0> dist[N];
}

parameters {
  real<lower = 0> sigma;
}

model {
  real p[N];

  for (n in 1:N) {
    p[n] <- 2 * Phi(1 / sigma * asin((R - r) / dist[n]) ) - 1;
  }


  successes ~ binomial(tries, p);
}
```

```
data {
  int N;
  int<lower = 0> tries[N];
  int<lower = 0> successes[N];
  real<lower = 0> dist[N];
}

parameters {
  real<lower = 0> sigma;
}

model {
  real p[N];

  for (n in 1:N) {
    p[n] <- 2 * Phi(1 / sigma * asin((R - r) / dist[n]) ) - 1;
  }

  sigma ~ cauchy(0, 2.5);
  successes ~ binomial(tries, p);
}
```

RUN IT

```
fit <- stan("golf.stan",
          data = list(N, tries, successes, dist))
```

What's wrong??

```
data {
  int N;
  int<lower = 0> tries[N];
  int<lower = 0> successes[N];
  real<lower = 0> dist[N];
}
transformed data {
  real R;
  real r;
  R <- 4.25 / 2;
  r <- 1.68 / 2;
}
parameters {
  real<lower = 0> sigma;
}
model {
  real p[N];

  for (n in 1:N) {
    p[n] <- 2 * Phi(1 / sigma * asin((R - r) / dist[n]) ) - 1;
  }

  sigma ~ cauchy(0, 2.5);
  successes ~ binomial(tries, p);
}
```

RUN IT

```
fit <- stan(“golf.stan”,
          data = list(N, tries, successes, dist))
```

What’s wrong??

The resulting estimate is $\hat{\sigma} = 0.026$ (which, when multiplied by $180/\pi$, comes out to $1.5°$);

RUN IT

```
dist <- dist * 12   ## convert to inches
```

```
fit <- stan("golf.stan",
            data = list(N, tries, successes, dist))
```

EXPLORE...

```
print(fit, digits = 3)

library(shinystan); launch_shinystan(fit)
```

FUNCTIONS
GENERATED QUANTITIES

ADD FUNCTIONS BLOCK FOR CALCULATING THETA0

GENERATED QUANTITIES FOR SIGMA IN DEGREES

GENERATED QUANTITIES FOR POSTERIOR ESTIMATES OF MADE PUTTS

RECAP

RECAP

- Wrote Stan programs
- Fit Stan programs
- Debugged Stan programs
- Overall: replication

THANKS

- Help
  - http://mc-stan.org
  - stan-users mailing list
- Stan Group Inc.
  http://stan.fit
  training / statistical support / consulting
- bearlee@alum.mit.edu / @djsyclik / @mcmc_stan
  eric@stan.fit / @ericnovik

A Fiscally Sponsored Project of
NUMFOCUS
OPEN CODE = BETTER SCIENCE