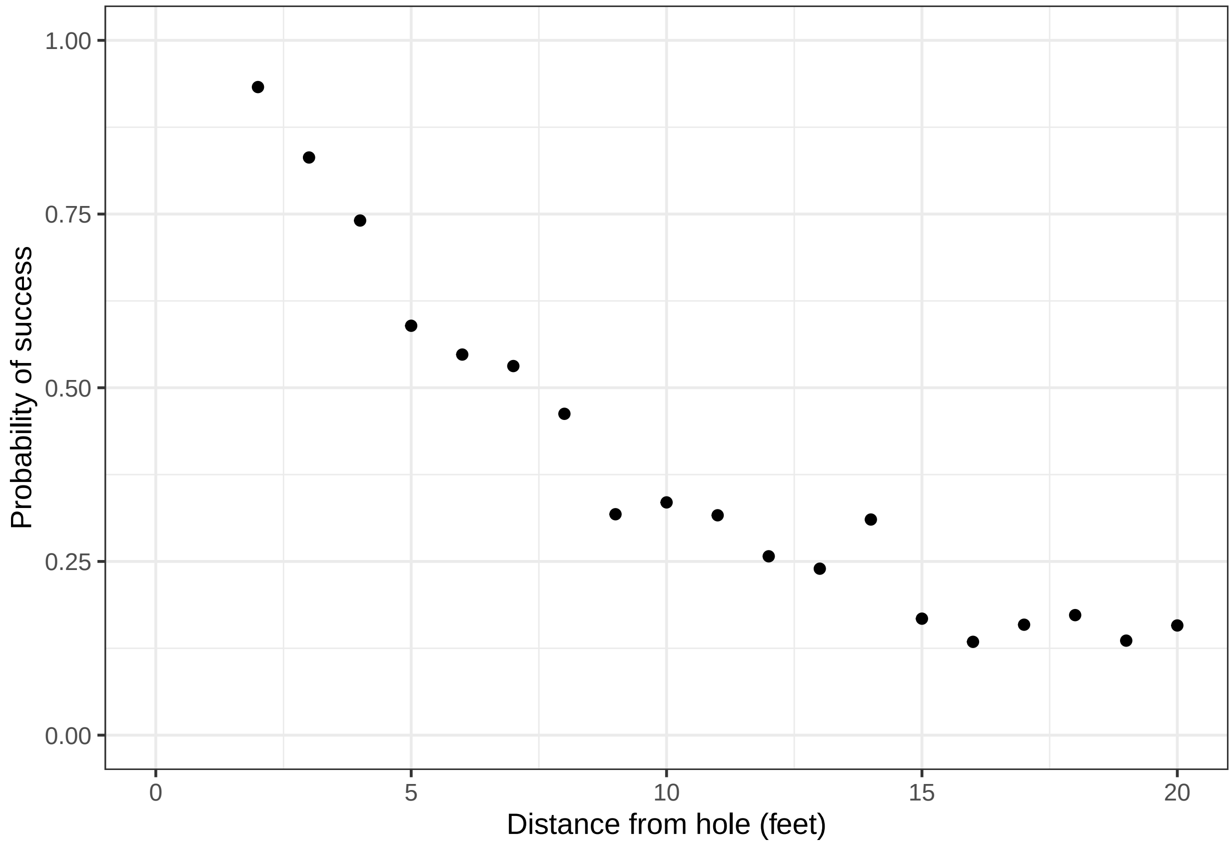
Model building and expansion example: Golf putting

Input data

Data from professional golfers on the proportion of successful putts as a function of distance from the hole:

- x is the distance from the hole (feet)
- n is the number of attemps
- y is the number of successful shots made
- $p = \frac{y}{n}$ is the probability of success

```
## # A tibble: 19 x 4
##        x     n     y     p
##    <dbl> <dbl> <dbl> <dbl>
##  1     2  1443  1346 0.933
##  2     3   694   577 0.831
##  3     4   455   337 0.741
##  4     5   353   208 0.589
##  5     6   272   149 0.548
##  6     7   256   136 0.531
##  7     8   240   111 0.462
##  8     9   217    69 0.318
##  9    10   200    67 0.335
## 10    11   237    75 0.316
## 11    12   202    52 0.257
## 12    13   192    46 0.240
## 13    14   174    54 0.310
## 14    15   167    28 0.168
## 15    16   201    27 0.134
## 16    17   195    31 0.159
## 17    18   191    33 0.173
## 18    19   147    20 0.136
## 19    20   152    24 0.158
```



First model: Logistic regression

A natural first model is logistic regression:

$$y_j \sim \text{binomial}(n_j, p_j), \text{ for } j = 1, \ldots, J p_j = \text{logit}^{-1}(a + bx_j) = 1/(1 + e^{a+bx_j})$$

with stan model:

```
// J observations of n (trials), y (successes) for each distance value x
data {
  int<lower=0> J;
  int n[J];
  int y[J];
  vector[J] x;
}

// Parametrised by a and b
parameters {
  real a;
  real b;
}

model {
  vector[J] p;

  for (j in 1:J) {
```

```
    p[j] = inv_logit(a + b*x[j]);
  }

  y ~ binomial(n, p);
}

generated quantities {
  vector[J] y_rep;
  vector[J] p;

  for (j in 1:J) {
    p[j] = inv_logit(a + b*x[j]);
    y_rep[j] = binomial_rng(n[j], p[j]);
  }
}
```

where uniform priors are used due to the high number of trials for each n_j.

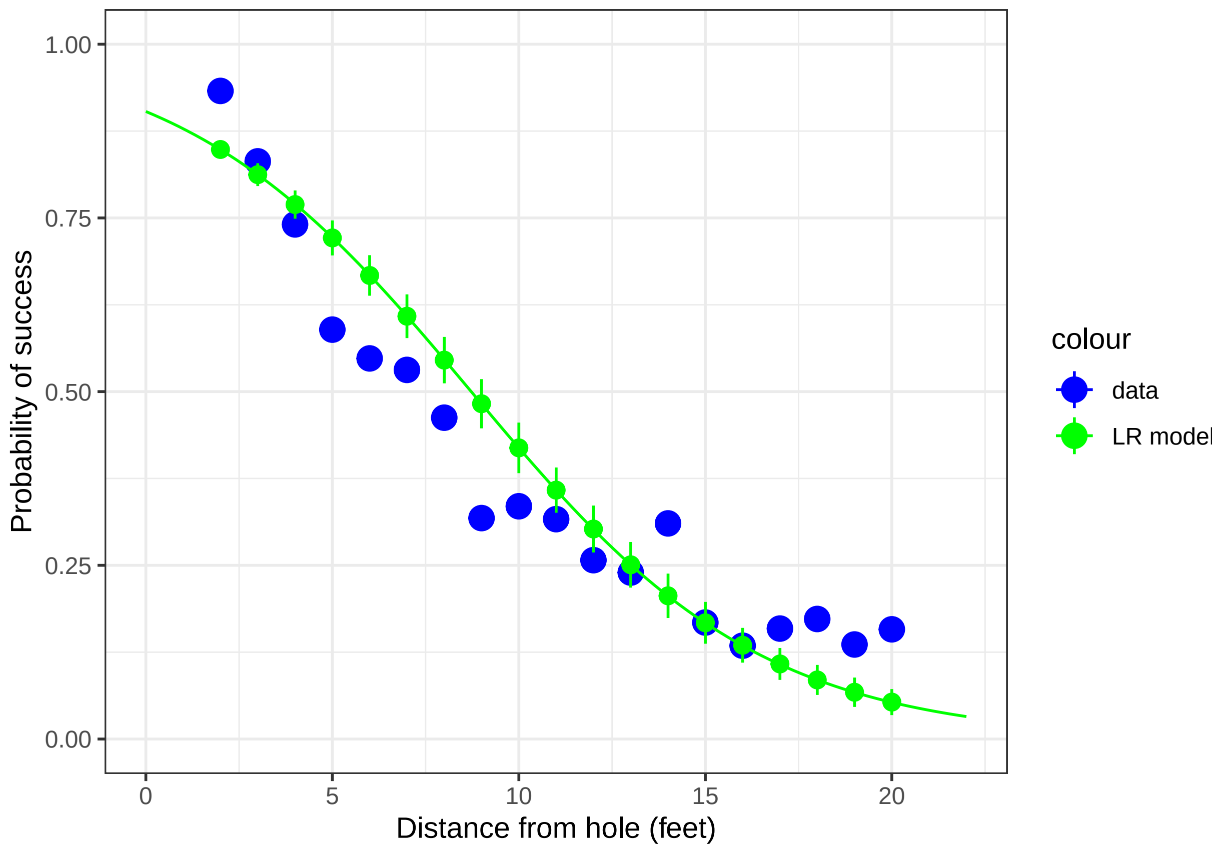
```
##
## SAMPLING FOR MODEL '7a14f0a730dda1e47904c69d4dbd8eed' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.045 seconds (Warm-up)
## Chain 1:                0.037 seconds (Sampling)
## Chain 1:                0.082 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL '7a14f0a730dda1e47904c69d4dbd8eed' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
```

```
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.041 seconds (Warm-up)
## Chain 2:                0.039 seconds (Sampling)
## Chain 2:                0.08 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL '7a14f0a730dda1e47904c69d4dbd8eed' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.044 seconds (Warm-up)
## Chain 3:                0.039 seconds (Sampling)
## Chain 3:                0.083 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL '7a14f0a730dda1e47904c69d4dbd8eed' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
```

```
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.04 seconds (Warm-up)
## Chain 4:                0.042 seconds (Sampling)
## Chain 4:                0.082 seconds (Total)
## Chain 4:
```

We can posterior

```
## # A tibble: 1 x 2
##       a      b
##   <dbl>  <dbl>
## 1  2.23 -0.256
```



Second model: Considering geometry

The figure below shows a simplified sketch of a golf shot:

The dotted lines represent the (maximum) angle which the ball of radius r must be hit so that it falls within the hole of radius R, being hit from a distance of x feet. This maximum angle is:

$$\sin^{-1}\left(\frac{R-r}{x}\right)$$

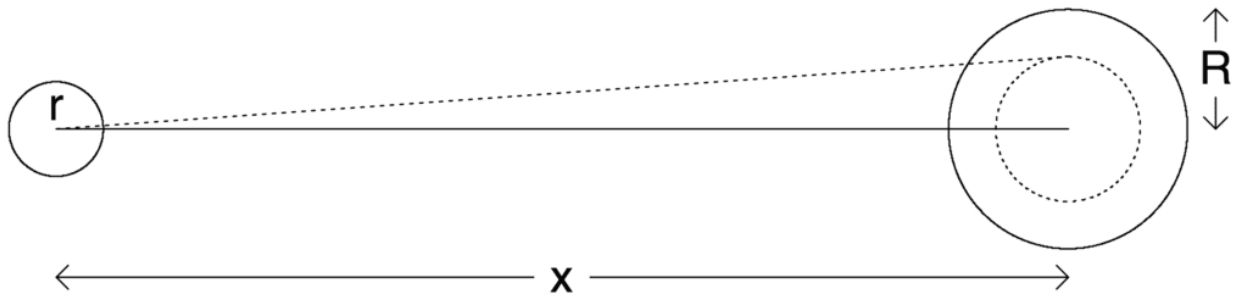


Figure 1: Simple geometrical model of golf putting.

We assume that the golfer attempts to hit the ball straight but that many small factors interfere with this goal. Hence, we assume the actual angle k follows a normal distribution centered at 0 with some standard deviation σ.

The probability that the ball goes inside the all is equal to the probability that the angle is below the maximum threshold, that is,

$$\Pr\Big(k < \sin^{-1}\big((R-r)/x\big)\Big) = 2\Phi\left(\frac{\sin^{-1}\big((R-r)/x\big)}{\sigma}\right) - 1$$

transforming the original model into:

$$y_j \sim \text{binomial}(n_j, p_j), \text{ for } j = 1, \ldots, J p_j = 2\Phi\left(\frac{\sin^{-1}\big((R-r)/x\big)}{\sigma}\right) - 1$$

with stan model:

```
// J observations of n (trials), y (successes) for each distance value x
data {
  int<lower=0> J;
  int n[J];
  int y[J];
  vector[J] x;
  real r;
  real R;
}

// Parametrised by a and b
parameters {
  real sigma;
}

model {
  vector[J] p;

  for (j in 1:J) {
    p[j] = 2*Phi(asin((R-r)/x[j]) / sigma) - 1;
  }

  y ~ binomial(n, p);
}
```

```
generated quantities {
  real sigma_degrees;
  vector[J] y_rep;
  vector[J] p;

  sigma_degrees = (180/pi())*sigma;

  for (j in 1:J) {
    p[j] = 2*Phi(asin((R-r)/x[j])/sigma) - 1;
    y_rep[j] = binomial_rng(n[j], p[j]);
  }
}
```

```
r_feet <- 1.68/2 * 0.0833 # inches to feet
R_feet <- 4.25/2 * 0.0833 # inches to feet

fit2 <- stan(
  model_code = model2,
  chains = 4,
  iter = 2000,
  data = list(
    J = nrow(golf), x = golf$x, y = golf$y, n = golf$n, r = r_feet, R = R_feet
  )
)
```

```
##
## SAMPLING FOR MODEL '84a471691ae3368701e4cc7915f5af4e' NOW (CHAIN 1).
## Chain 1: Rejecting initial value:
## Chain 1:   Error evaluating the log probability at the initial value.
## Chain 1: Exception: binomial_lpmf: Probability parameter[1] is -0.0622276, but must be in the interva
##
## Chain 1: Rejecting initial value:
## Chain 1:   Error evaluating the log probability at the initial value.
## Chain 1: Exception: binomial_lpmf: Probability parameter[1] is -0.0473697, but must be in the interva
##
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
```

```
## Chain 1:  Elapsed Time: 0.026 seconds (Warm-up)
## Chain 1:                0.024 seconds (Sampling)
## Chain 1:                0.05 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL '84a471691ae3368701e4cc7915f5af4e' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.025 seconds (Warm-up)
## Chain 2:                0.023 seconds (Sampling)
## Chain 2:                0.048 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL '84a471691ae3368701e4cc7915f5af4e' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.028 seconds (Warm-up)
## Chain 3:                0.021 seconds (Sampling)
## Chain 3:                0.049 seconds (Total)
## Chain 3:
```

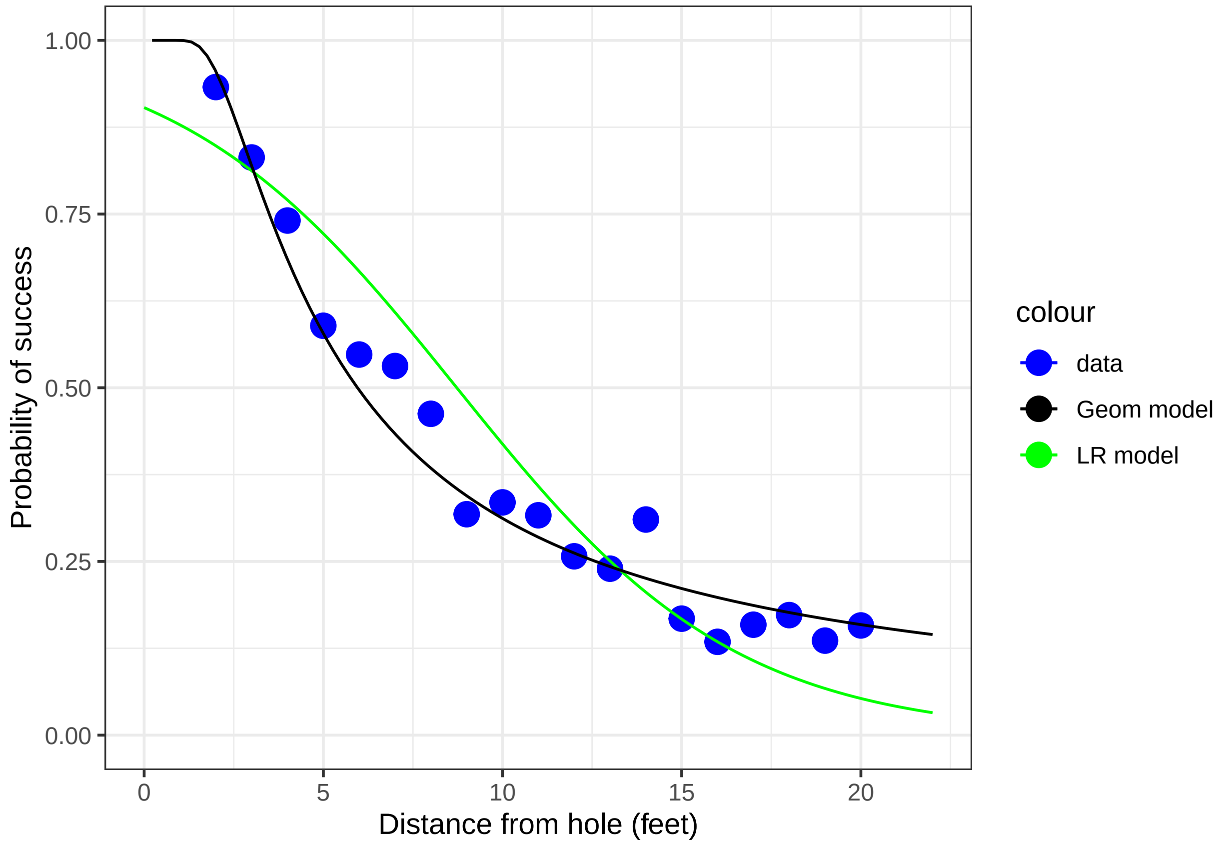
```
##
## SAMPLING FOR MODEL '84a471691ae3368701e4cc7915f5af4e' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.026 seconds (Warm-up)
## Chain 4:                0.022 seconds (Sampling)
## Chain 4:                0.048 seconds (Total)
## Chain 4:
```

We can posterior

```
## # A tibble: 1 x 2
##    sigma sigma_degrees
##    <dbl>         <dbl>
## 1 0.0267          1.53
```

```
## Warning in asin((R - r)/x): NaNs produced
```

```
## Warning: Removed 1 row containing missing values (`geom_function()`).
```



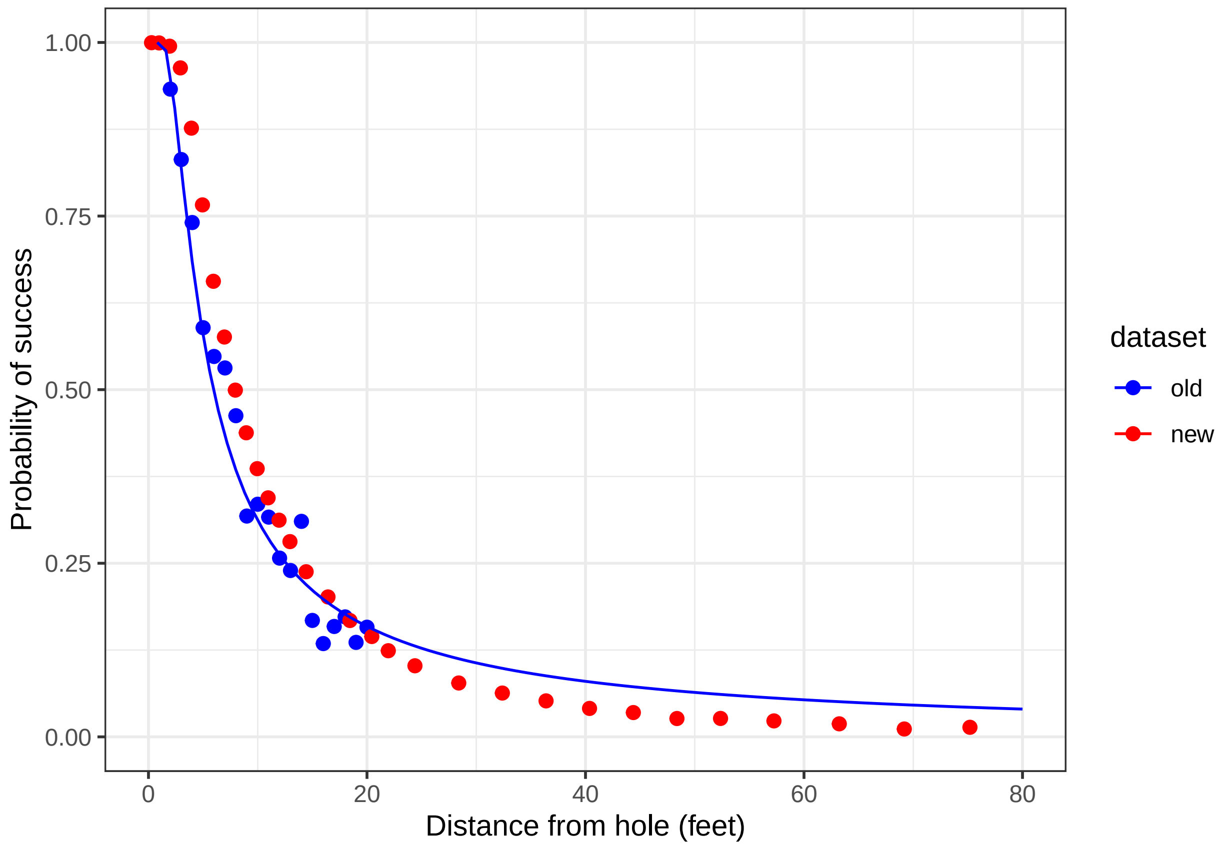
New golf data

```
## # A tibble: 31 x 3
##         x      n      y
##     <dbl>  <dbl>  <dbl>
##  1  0.28  45198  45183
##  2  0.97 183020 182899
##  3  1.93 169503 168594
##  4  2.92 113094 108953
##  5  3.93  73855  64740
##  6  4.94  53659  41106
##  7  5.94  42991  28205
##  8  6.95  37050  21334
##  9  7.95  33275  16615
## 10  8.95  30836  13503
## 11  9.95  28637  11060
## 12 11.0   26239   9032
## 13 12.0   24636   7687
## 14 13.0   22876   6432
## 15 14.4   41267   9813
## 16 16.4   35712   7196
## 17 18.4   31573   5290
## 18 20.4   28280   4086
## 19 22.0   13238   1642
## 20 24.4   46570   4767
## 21 28.4   38422   2980
```

```
## 22 32.4   31641   1996
## 23 36.4   25604   1327
## 24 40.4   20366    834
## 25 44.4   15977    559
## 26 48.4   11770    311
## 27 52.4    8708    231
## 28 57.2    8878    204
## 29 63.2    5492    103
## 30 69.2    3087     35
## 31 75.2    1742     24
```

```
## Warning in asin((R - r)/x): NaNs produced
```

```
## Warning: Removed 1 row containing missing values (`geom_function()`).
```



Geometrical model on new data over-predicts the probability of success for longer shots (distance greater than 20 feet) and under-predicts the probability of success for putts under 20 feet.

These differences are perhaps due to measurement error (distance is more precisely measured on new data and probably rounded up in old data) or increase in player performance over the years.

Model 3: Accounting for hitting force

The angle is not the only factor that a golfer must control for when hitting the ball; it also needs to hit the ball with the right amount of force.

A second parameter is introduced to account for the golfer's control over distance. Suppose u indicates how far the ball travels when hit. Then, a ball goes in if (a) the angle is within the threshold and (b) u is in range $[x, x+3]$ (the ball is hit hard enough to arrive at the hole but not hard enough that it would go too far).

Broadie's model assumes that a golfer will try to hit the ball one foot past the hole but with a multiplicative error in the shot's potential distance, so that $u = (x+1) \cdot (1+\epsilon)$, where the error term ϵ is normally distributed with mean 0 and standard deviation σ_{distance}.

Formally the model for u is defined as:

$$u \sim \text{Normal}(x+1, (x+1))$$

Model 4: Expanding model 3 with a fudge factor