Making a function in Python

```
def function_name(arguments):
    expression
    return variable_name


def diff_two_nums(x, z):
    y = x - z
    return y
}
```

Py

Vectorized programming

Demo in R

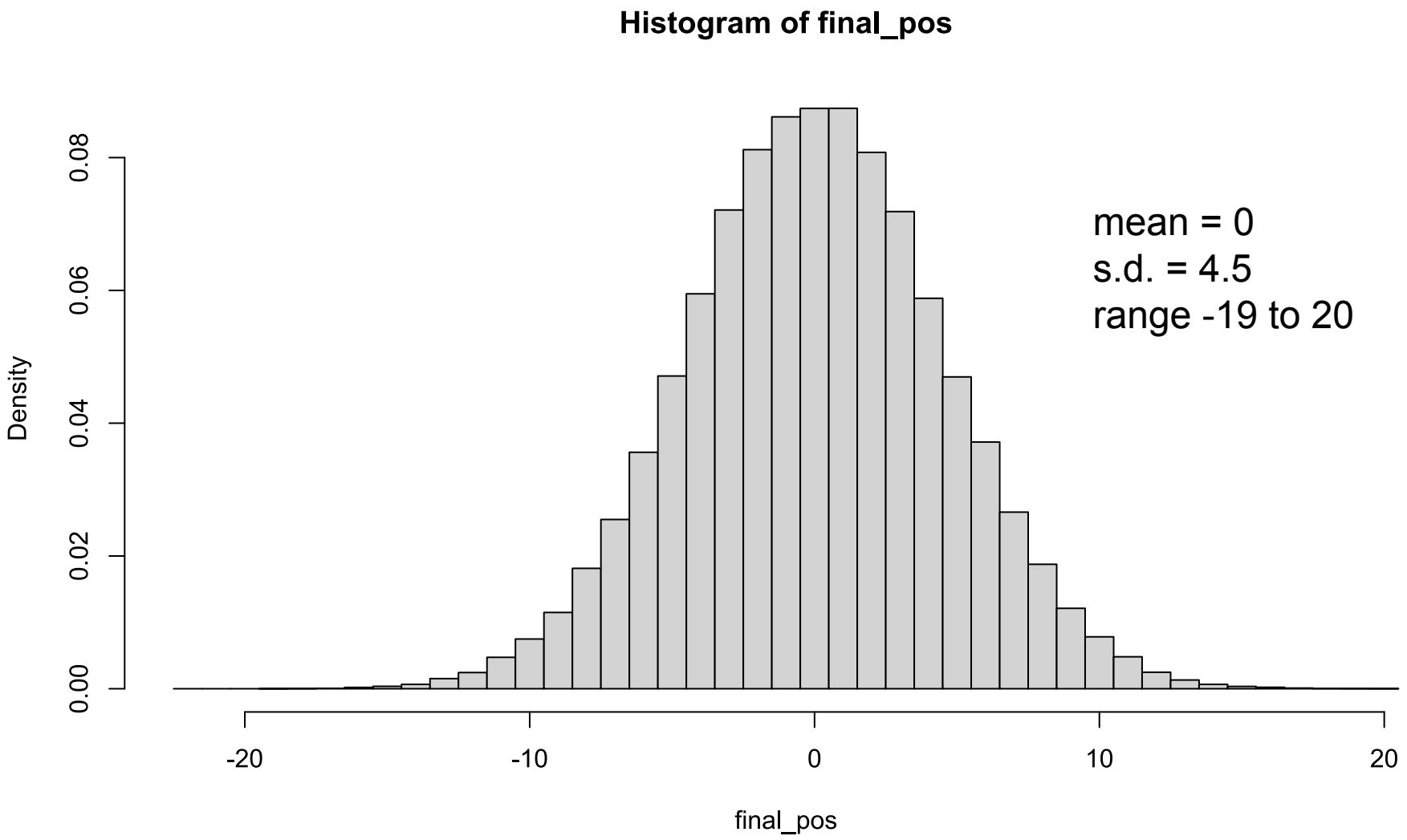
Vectorized programming

Demo in R

Data generating process

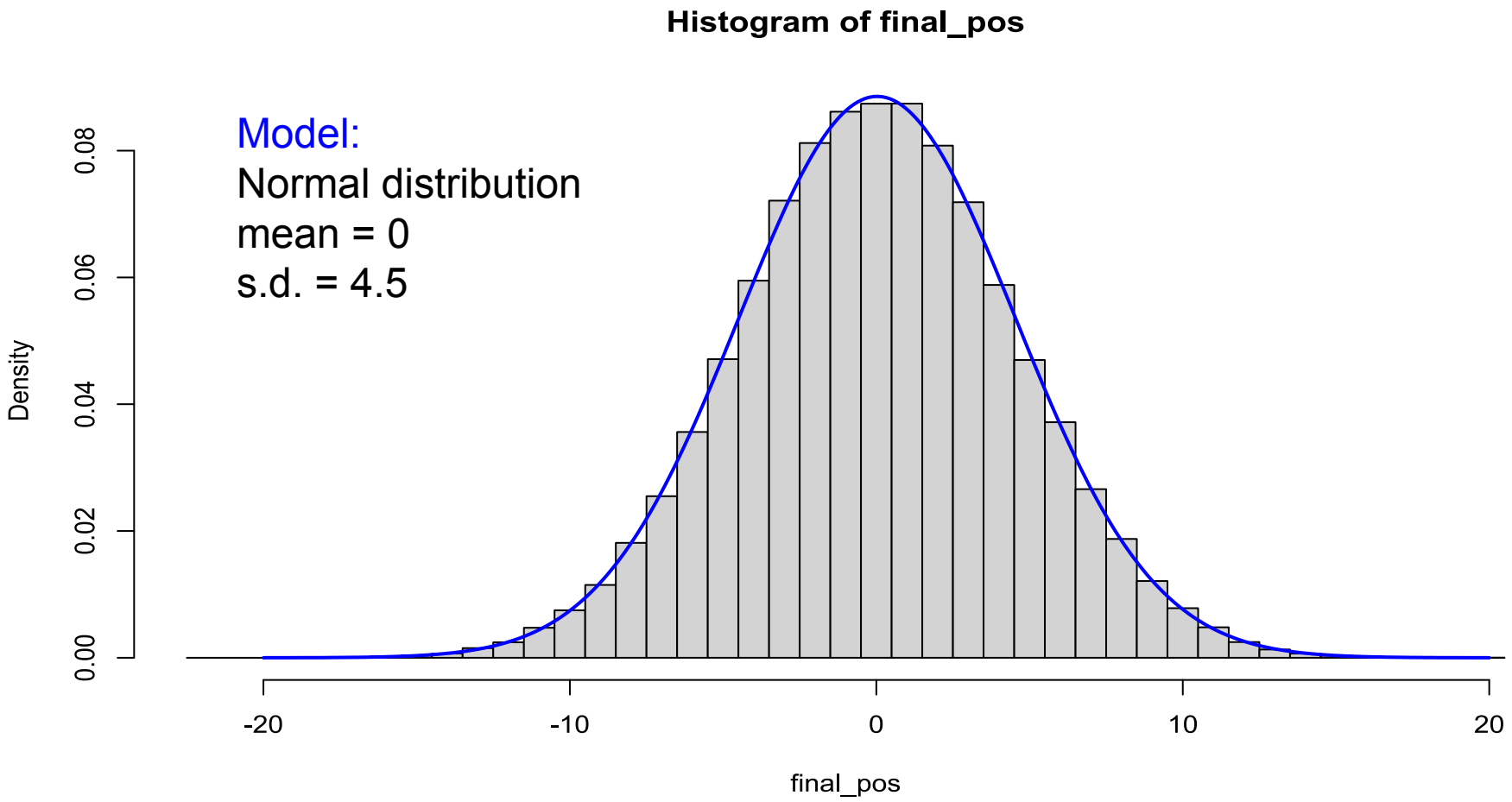
- We've seen biological models:
- Deterministic models (house finch)
- Stochastic DGP:
  - Movement, finding nut
  - Intrinsic stochastic process
- Next: Deterministic skeleton + noise

Generated data



A new data generating model

Phenomenological scale of abstraction



Data generating process

- Next: Deterministic skeleton + noise
- e.g. could use with house finch
- But first, descriptive models
  - e.g. linear normal model

Make a function

```
function_name <- function(arguments) {
    expression
    return(object)
}
```

Exercise:
Make a function to calculate the linear model given the model parameters and a vector of x data. In other words, turn the following into a function:

```
y <- b_0 + b_1 * x
```

Use vectorized operations

R

Make a function

```
function_name <- function(arguments) {
    expression
    return(object)
}
```

Exercise:
Make a function to calculate the linear model given the model parameters and a vector of x data. In other words, turn the following into a function:

```
y <- b_0 + b_1 * x
```

Solution:

```
lin_skel <- function(b_0, b_1, x) {
  y <- b_0 + b_1 * x
  return(y)
}
```

Use vectorized operations

R

Make it a stochastic DGP

```
rnorm(n, mean, sd)
```

Exercise:
Make a function to generate normal random data from the linear deterministic skeleton

R

How in Python?

- Libraries
  - Conda package manager
  - Many others (pip, ...)
- Numpy library
- Data structure: Numpy arrays
- Vectorized operations
- Matplotlib library