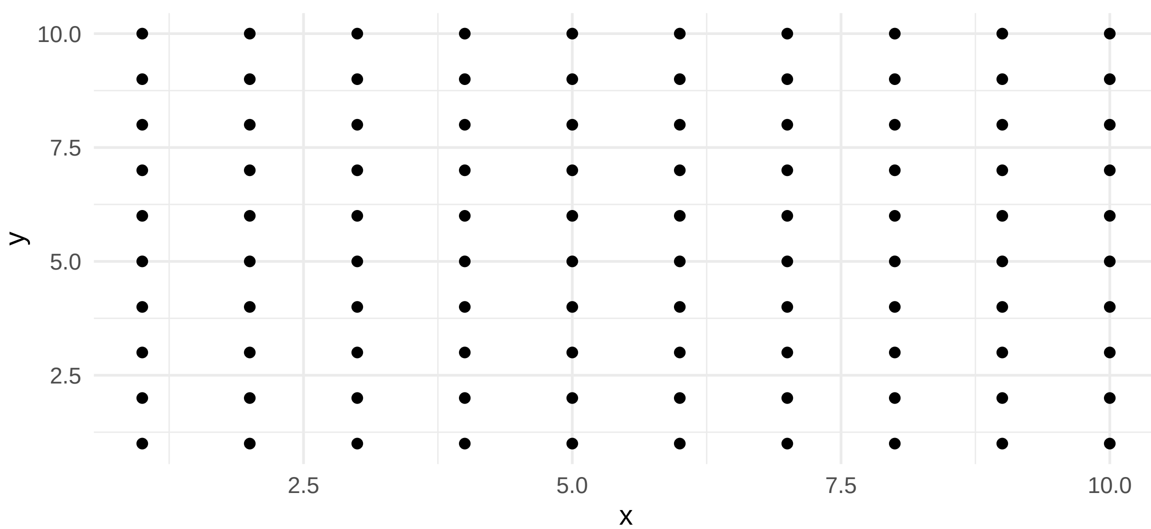
John Hendley & Randy Nguyen

STA 6375

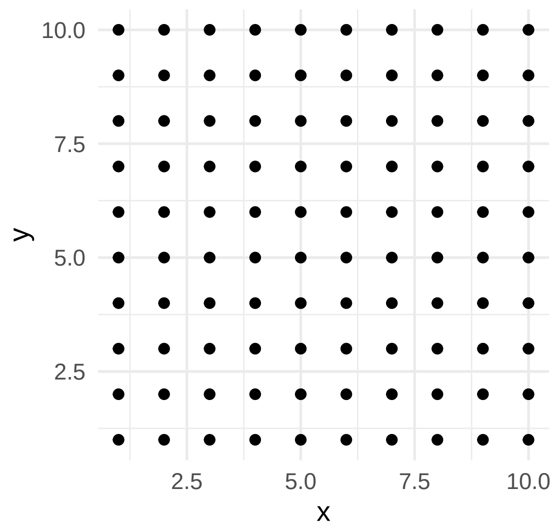
Homework 3

Question 1

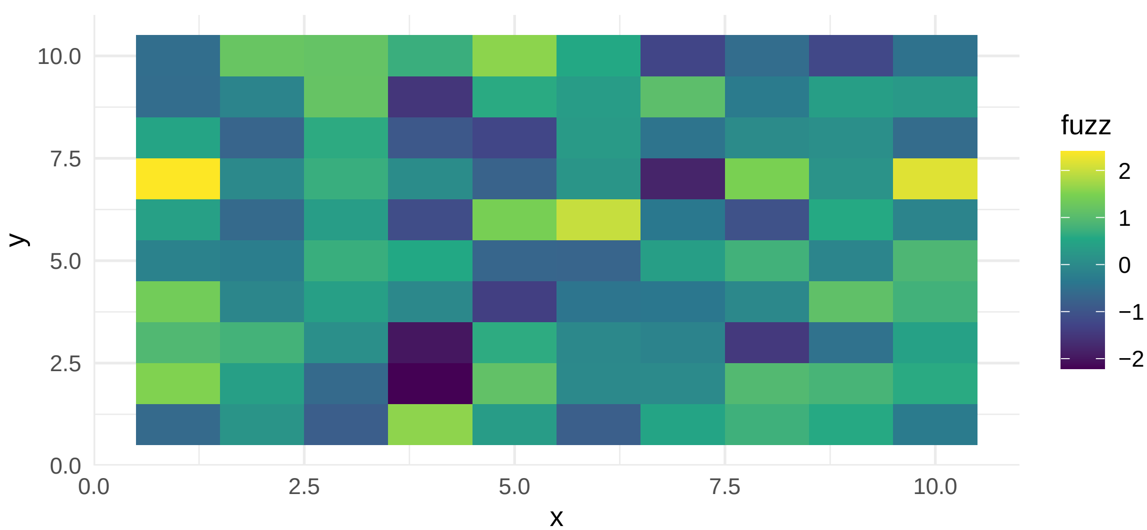
a.
```r
library("tidyverse")
# Calling the data frame df
df <- expand.grid("x" = 1:10, "y" = 1:10)
ggplot(df, aes(x, y)) +
  geom_point() +
  theme_minimal()
```



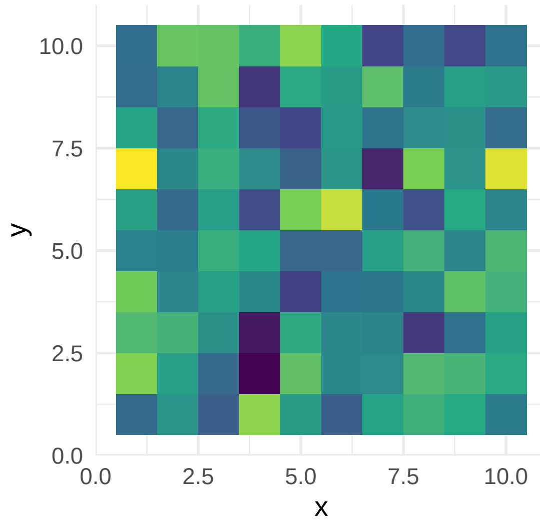
b.
```r
ggplot(df, aes(x, y)) +
  geom_point() +
  theme_minimal() +
  coord_equal()
```



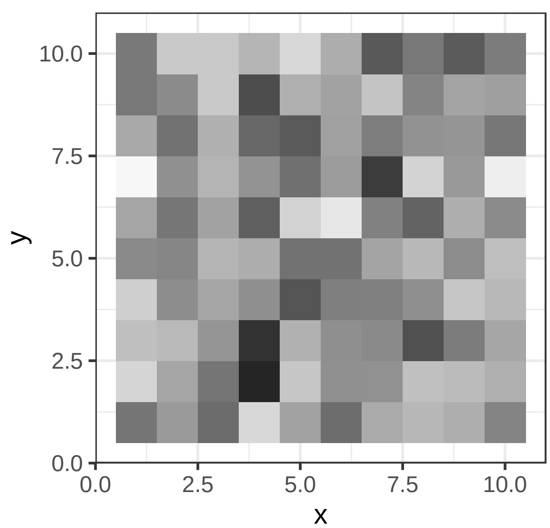
c.
```r
set.seed(1)
fuzz <- rnorm(nrow(df))
ggplot(df, aes(x, y, fill = fuzz)) +
  theme_minimal() +
  geom_tile() # looks better than geom_bin2d
```



d.
```r
set.seed(1)
fuzz <- rnorm(nrow(df))
ggplot(df, aes(x, y, fill = fuzz)) +
  theme_minimal() +
  geom_tile() +
  theme(legend.position = "none") +
  coord_equal()
```

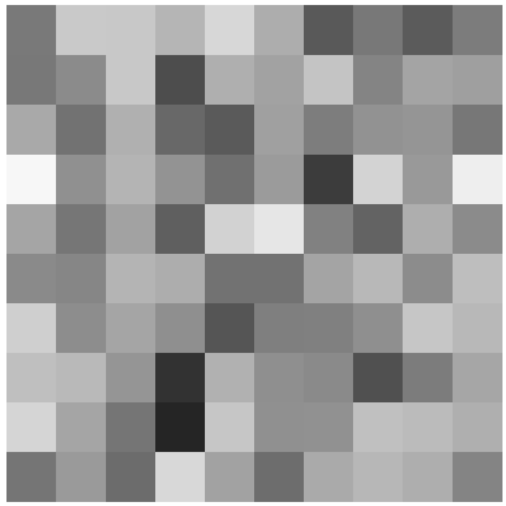


e.
```r
set.seed(1)
fuzz <- rnorm(nrow(df))
ggplot(df, aes(x, y, fill = fuzz)) +
  theme_bw() +
  geom_tile() +
  coord_equal() +
  theme(legend.position = "none") +
  scale_fill_distiller(palette = "Greys")
```

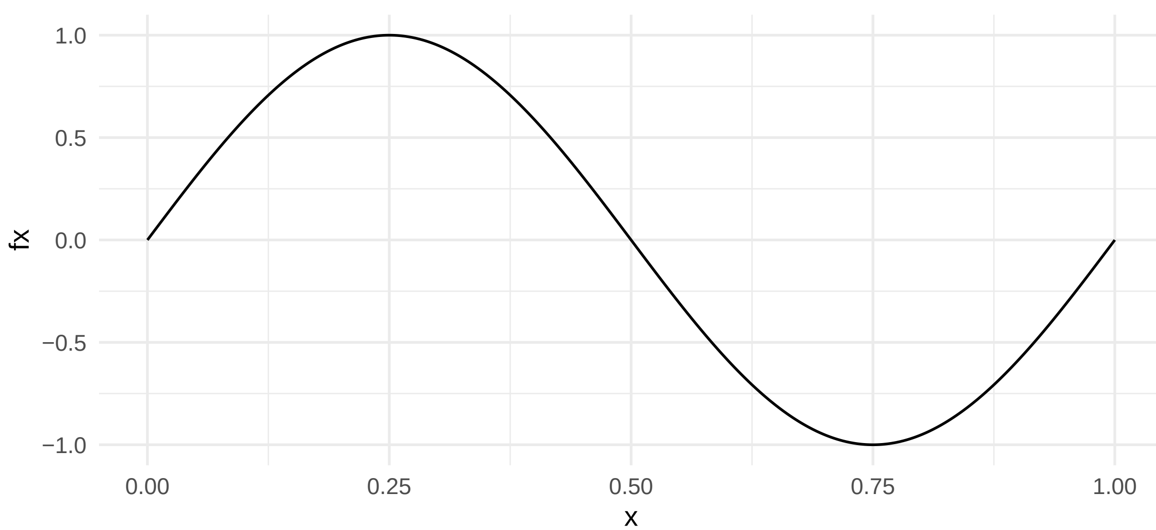


f.
```r
set.seed(1)
fuzz <- rnorm(nrow(df))
ggplot(df, aes(x, y, fill = fuzz)) +
  geom_tile() +
  coord_equal() +
  scale_fill_distiller(palette = "Greys") +
  ylab(NULL) +
```

```r
  xlab(NULL) +
  theme_void() +
  theme(legend.position = "none")
```

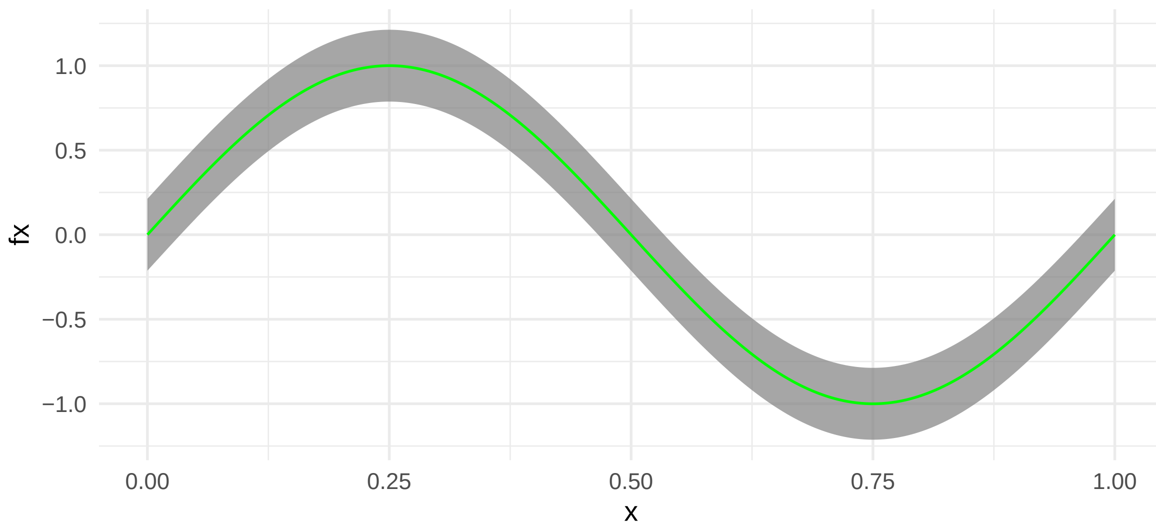


g.
```r
x <- seq(0, 1, 1e-4)
fx <- sin(2*pi*x)
sine <- data.frame("x" = x, "y" = fx)
ggplot(sine, aes(x, fx)) +
  theme_minimal() +
  geom_line()
```



h.
```r
x <- seq(0, 1, 1e-4)
fx <- sin(2*pi*x)
sine <- data.frame("x" = x, "y" = fx)
ggplot(sine, aes(x, fx)) +
```

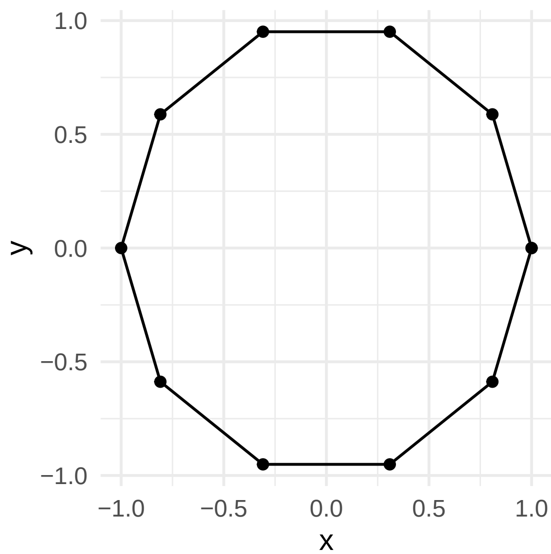
```r
  theme_minimal() +
  # Can't tell the exact width of the shaded grey region
  geom_ribbon(aes(ymin = fx - 0.2125, ymax = fx + 0.2125), fill = "grey50", alpha = 0.7) +
  geom_line(color = "green")
```



i.
```r
# Calculating the coordinates equally spaced
x <- c()
y <- c()

for (i in 0:10) {
  x <- c(x, cos(i * pi/5))
  y <- c(y, sin(i * pi/5))
}

decagon <- data.frame(x, y)
ggplot(decagon, aes(x, y)) +
  geom_point() +
  geom_path() +
  theme_minimal()
```



j.
```r
set.seed(1)
df <- data.frame(x = rnorm(1e3, mean = 3, sd = 2))
ggplot(df, aes(x)) +
  geom_density() +
  stat_function(fun = dnorm, args = list(mean = 3, sd = 2), color = "red") +
  geom_vline(xintercept = 8, color = "red", linetype = "twodash", size = 1) +
  theme_minimal()
```

Question 2

a.
```r
A <- matrix(c(
  -1, 3, 1,
  -7, 9, 1,
  -2, 3, 4),
```

```r
  nrow = 3, byrow = TRUE)

r <- eigen(A)

# V
(V <- r$vector)
```

```
##            [,1]      [,2]       [,3]
## [1,] -0.3796421 0.3574067  0.6785983
## [2,] -0.6749193 0.3574067  0.6785983
## [3,] -0.6327368 0.8628562 -0.2810846
```

```r
# Eigenvalues
(lam <- r$values)
```

```
## [1] 6.000000 4.414214 1.585786
```

```r
# Diagonal matrix Lambda
Lambda <- diag(lam, nrow = 3, ncol = 3)

# Verification
V %*% Lambda %*% solve(V)
```

```
##      [,1] [,2] [,3]
## [1,]   -1    3    1
## [2,]   -7    9    1
## [3,]   -2    3    4
```

b.

```r
A <- matrix(c(
  10, 2, -6,
  2, 7, 0,
  -6, 0, 2),
  nrow = 3, byrow = TRUE)

r <- eigen(A)

# Eigenvectors
(V <- r$vector)
```

```
##            [,1]       [,2]       [,3]
## [1,]  0.8595576 -0.1725817  0.4810159
## [2,]  0.2573243  0.9593838 -0.1156155
## [3,] -0.4415258  0.2231553  0.8690551
```

```r
# Eigenvalues
(lam <- r$values)
```

```
## [1] 13.680735  6.640224 -1.320958
```

```r
# Diagonal matrix Lambda
Lambda <- diag(lam, nrow = 3, ncol = 3)

# V is orthogonal
zapsmall(crossprod(V))
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    1    0
## [3,]    0    0    1
```

```r
# Verification
zapsmall(V %*% Lambda %*% t(V))
```

```
##      [,1] [,2] [,3]
## [1,]   10    2   -6
## [2,]    2    7    0
## [3,]   -6    0    2
```

c.
```r
A <- matrix(c(
  1, 5, 6,
  2, 6, 8,
  3, 7, 10,
  4, 8, 12),
  nrow = 4, byrow = TRUE)

s <- svd(A, nu = 4)

# U 4x4 matrix
s$u
```

```
##            [,1]       [,2]       [,3]       [,4]
## [1,] -0.3340803 -0.7670661  0.5425798 -0.0748813
## [2,] -0.4359333 -0.3316054 -0.6676264  0.5042568
## [3,] -0.5377863  0.1038552 -0.2924864 -0.7838697
## [4,] -0.6396393  0.5393158  0.4175331  0.3544942
```

```r
# s$u is orthogonal
zapsmall(s$u %*% t(s$u))
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    1    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
```

```r
# V 3x3 matrix
s$v
```

```
##            [,1]       [,2]       [,3]
## [1,] -0.2301002  0.7834032  0.5773503
## [2,] -0.5633970 -0.5909742  0.5773503
## [3,] -0.7934972  0.1924290 -0.5773503
```

```r
# s$v is orthogonal
zapsmall(s$v %*% t(s$v))
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    1    0
## [3,]    0    0    1
```

```r
# 4x3 matrix Sigma, including the zero entries
(D <- diag(s$d, nrow = dim(s$u)[1], ncol = dim(s$v)[2]))
```

```
##          [,1]     [,2]         [,3]
## [1,] 23.37183 0.000000 0.000000e+00
## [2,]  0.00000 1.325693 0.000000e+00
## [3,]  0.00000 0.000000 9.287939e-16
## [4,]  0.00000 0.000000 0.000000e+00
```

```r
# Verification
zapsmall(s$u %*% D %*% t(s$v))
```

```
##      [,1] [,2] [,3]
## [1,]    1    5    6
## [2,]    2    6    8
## [3,]    3    7   10
## [4,]    4    8   12
```

d.
```r
(A <- matrix(1:4, nrow = 2)) # A is invertible
```

```
##      [,1] [,2]
## [1,]    1    3
## [2,]    2    4
```

```r
(elu <- Matrix::expand(Matrix::lu(A)))
```

```
## $L
## 2 x 2 Matrix of class "dtrMatrix" (unitriangular)
##      [,1] [,2]
## [1,]  1.0    .
## [2,]  0.5  1.0
##
## $U
## 2 x 2 Matrix of class "dtrMatrix"
##      [,1] [,2]
## [1,]    2    4
## [2,]    .    1
##
## $P
## 2 x 2 sparse Matrix of class "pMatrix"
##
## [1,] . |
## [2,] | .
```

```r
# Verification
with(elu, P %*% L %*% U)
```

```
## 2 x 2 Matrix of class "dgeMatrix"
##      [,1] [,2]
## [1,]    1    3
## [2,]    2    4
```

e.
```r
A <- matrix(c(
  4, 2, 1,
  2, 4, 2,
  1, 2, 4),
  nrow = 3, byrow = TRUE)

# A is a square 3x3 matrix, and
# A is a symmetric matrix positive definite matrix since
# the entries are positive and a_{ij} = a_{ji} for all i and j
# so A satisfies the conditions for the Cholesky decomposition

(U <- chol(A))
```

```
##      [,1]     [,2]      [,3]
## [1,]    2 1.000000 0.5000000
## [2,]    0 1.732051 0.8660254
## [3,]    0 0.000000 1.7320508
```

```r
# Verification
crossprod(U)
```

```
##      [,1] [,2] [,3]
## [1,]    4    2    1
## [2,]    2    4    2
## [3,]    1    2    4
```

f.
```r
A <- matrix(c(
  1, 3, 2,
  3, 0, 0,
  0, 1, 3,
  0, 1, 0),
  nrow = 4, byrow = TRUE)

# U
(U <- qr.R(qr(A)))
```

```
##           [,1]       [,2]       [,3]
## [1,] -3.162278 -0.9486833 -0.6324555
## [2,]  0.000000  3.1780497  2.6431305
## [3,]  0.000000  0.0000000 -2.3693589
```

```r
# Q (R)
(Q <- qr.Q(qr(A)))
```

```
##            [,1]       [,2]        [,3]
## [1,] -0.3162278  0.8495776  0.18804435
## [2,] -0.9486833 -0.2831925 -0.06268145
## [3,]  0.0000000  0.3146584 -0.91514919
## [4,]  0.0000000  0.3146584  0.35101613
```

```r
# Q is orthogonal
zapsmall(crossprod(Q))
```

```
##      [,1] [,2] [,3]
## [1,]    1    0    0
## [2,]    0    1    0
## [3,]    0    0    1
```

```r
# Verification
zapsmall(Q %*% U)
```

```
##      [,1] [,2] [,3]
## [1,]    1    3    2
## [2,]    3    0    0
## [3,]    0    1    3
## [4,]    0    1    0
```