Pr. 1.
Prove how the **eigenvalues** and **eigenvectors** of $\boldsymbol{B} = \boldsymbol{A} - 10\boldsymbol{I}$ relate to the eigenvalues and eigenvectors of $\boldsymbol{A}$, when $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{F}^{N \times N}$. (Show both directions: from $\boldsymbol{A}$ to $\boldsymbol{B}$ and vice versa.)

Pr. 2.
Let $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_N$ denote **orthonormal** vectors in $\mathbb{R}^N$. Show that $\boldsymbol{A}\boldsymbol{v}_1, \boldsymbol{A}\boldsymbol{v}_2, \ldots, \boldsymbol{A}\boldsymbol{v}_N$ are also orthonormal vectors if and only if $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ is an orthogonal matrix.

Pr. 3.
The **Frobenius norm** of a $M \times N$ matrix $\boldsymbol{A}$ is defined as $\|\boldsymbol{A}\|_{\mathrm{F}} = \sqrt{\sum_{i,j} |a_{ij}|^2} = \sqrt{\sum_{i=1}^M \sum_{j=1}^N |a_{ij}|^2}$. Express the Frobenius norm of $\boldsymbol{A}$ in terms of its **singular values**. Hint: First express $\|\boldsymbol{A}\|_{\mathrm{F}}$ in terms of the matrix $\boldsymbol{A}'\boldsymbol{A}$.

Pr. 4.
Let $\boldsymbol{A}$ be an $M \times N$ matrix with **spectral norm** σ_1. Show the following bounds on the **Frobenius** norm:

$$\sigma_1 \leq \|\boldsymbol{A}\|_{\mathrm{F}} \leq \sqrt{\min(M, N)}\, \sigma_1.$$

The **operator norm** of a matrix equals its largest singular value σ_1. The norm of a matrix can be measured many ways. The above inequality shows that if the Frobenius norm is small then the operator norm is as well. However, the operator norm being small does not guarantee that the Frobenius norm will be as well. (Think for example the setting when M and N are very large).
Optional challenge: Is the upper bound **tight**?

Pr. 5.
The **Kronecker product** of two matrices $\boldsymbol{A} \in \mathbb{F}^{m_1 \times n_1}, \boldsymbol{B} \in \mathbb{F}^{m_2 \times n_2}$ is an important form of "**matrix multiplication**" defined as follows:

$$\boldsymbol{A} \otimes \boldsymbol{B} = \begin{bmatrix} a_{11}\boldsymbol{B} & a_{12}\boldsymbol{B} & \ldots & a_{1n}\boldsymbol{B} \\ & & \ddots & \\ a_{m1}\boldsymbol{B} & a_{m2}\boldsymbol{B} & \ldots & a_{mn}\boldsymbol{B} \end{bmatrix} \in \mathbb{F}^{(m_1 m_2) \times (n_1 n_2)}.$$

Let $\mathrm{vec}(\boldsymbol{C})$ denote the "vectorize" operation that returns as its output a vector in $\mathbb{F}^{mn}$ formed by stacking the n columns of $\boldsymbol{C} \in \mathbb{F}^{m \times n}$ on top of each other.
For arbitrary vectors $\boldsymbol{x} \in \mathbb{F}^M$ and $\boldsymbol{y} \in \mathbb{F}^N$, so that $\boldsymbol{x}\boldsymbol{y}^T \in \mathbb{F}^{M \times N}$, prove that

$$\mathrm{vec}(\boldsymbol{x}\boldsymbol{y}^T) = \boldsymbol{y} \otimes \boldsymbol{x} \in \mathbb{F}^{MN}.$$

Verify (for yourself) by using commands `vec(x * y')` and `kron(y, x)` in `Julia` for some real vectors, and using `vec(x * transpose(y))` for some complex vectors.
(The equality above does not hold for $\mathrm{vec}(\boldsymbol{x}\boldsymbol{y}')$ when $\boldsymbol{y}$ is complex.)

Pr. 6.
Use an **SVD** of a square matrix $\boldsymbol{A}$ to describe a **factorization** $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{S}$ where $\boldsymbol{Q}$ is unitary and $\boldsymbol{S} = \boldsymbol{S}' \succeq \boldsymbol{0}$, *i.e.*, $\boldsymbol{S}$ is **positive semidefinite**. Express $\boldsymbol{Q}$ and $\boldsymbol{S}$ in terms of the SVD components $\boldsymbol{U}$, $\boldsymbol{\Sigma}$ and $\boldsymbol{V}$ of $\boldsymbol{A}$.
Show that your $\boldsymbol{Q}$ and $\boldsymbol{S}$ satisfy the requirements.
(This matrix decomposition is analogous to the polar form $z = re^{i\theta}$ of a complex scalar z, where r is like $\boldsymbol{S}$ and $e^{i\theta}$ is like $\boldsymbol{Q}$.) Hint: $\boldsymbol{V}'\boldsymbol{V} = \boldsymbol{I}$.

Pr. 7.

This class uses the notation $\mathbb{F}$ to denote either the **field** $\mathbb{R}$ of real numbers or the **field** $\mathbb{C}$ of complex numbers. The purpose of this problem is to review what a field is. Refer to the appendix of the Ch. 1 notes as needed.
Determine whether the following sets (with the usual senses of multiplication, addition, etc.) are fields or not. If the answer is Yes then you may say so without proof. If the answer is No then give a concrete counter-example for one of the defining properties of a field that is violated.

(a) The set of numbers that are irrational or zero, *i.e.*, the set $(\mathbb{R} - \mathbb{Q}) \cup \{0\}$

(b) The set of $N \times N$ diagonal matrices (where the "1" element is $\boldsymbol{I}_N$, and the "0" element is $\mathbf{0}_{N\times N}$).

(c) The set of $N \times N$ diagonal matrices whose diagonal elements are either all zero or all non-zero.

(d) The set of **rational functions**, *i.e.*, functions of the form $P(x)/Q(x)$ where P and Q are both polynomials and Q is not zero.

(e) The set of $N \times N$ invertible matrices.

Pr. 8.

(a) Determine how **eigenvalues** and **eigenvectors** of

$$\boldsymbol{B} = \begin{bmatrix} \mathbf{0} & \boldsymbol{A} \\ \boldsymbol{A}' & \mathbf{0} \end{bmatrix},$$

are related to **singular values** and **singular vectors** of $\boldsymbol{A} \in \mathbb{F}^{N\times N}$. In other words, if $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}'$, then express eigenvalues and eigenvectors of $\boldsymbol{B}$ in terms of components of $\boldsymbol{U}$, $\boldsymbol{\Sigma}$ and $\boldsymbol{V}$.

Hints: Start with some numerical experiments using self-generated $\boldsymbol{A}$ matrices and form a conjecture.
The eigenvalues and singular values are intimately related. Keep in mind that $\boldsymbol{B}$ is $2N \times 2N$.
Think about combining $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ in various ways, such as $\boldsymbol{u}_i \pm \boldsymbol{v}_i$, $\begin{bmatrix} \pm\boldsymbol{u}_i \\ \pm\boldsymbol{v}_i \end{bmatrix}$, and $\begin{bmatrix} \pm\boldsymbol{v}_i \\ \pm\boldsymbol{u}_i \end{bmatrix}$.

(b) Optional: write a unitary eigendecomposition of $\boldsymbol{B}$ in matrix form.

This problem is related to one **SVD numerical approach**.

Pr. 9.

Your graduate school experience should be more than just taking courses. To convince you that we really mean that, you have the opportunity to earn 30 HW points by writing down some of your own non-course-related goals for this semester. Thinking beyond courses is especially important during the time of this pandemic, where self-care is crucial. In a few sentences or bullet points, list some of your personal non-course goals for this semester. The intent of this assignment is to encourage you to practice good self-care and encourage a growth-mindset.
Examples of activities someone might list would be: going on a walk in your neighborhood (remember social distancing) N times a week, making connections with N new people, attending N CSP seminars to learn about current research, exploring some topic beyond what is required in the course just for your own interest, mastering a new recipe, reading a novel, etc. Hopefully your list will include some things that will help you relax and/or think about the bigger picture. There will be another HW problem at the end of the semester where we will ask you to reflect on your goals for the semester. If what you did was different than your proposal, please tell us what you ended up doing. Obviously we won't know if you actually read that novel or not. This is really for you to be thinking about.

Pr. 10.
(Finite difference matrix)
This problem develops a tool that will be used in a later HW for an application called **photometric stereo**.
To approximate the derivatives of a function $f(x)$ that is sampled on the grid $x_1, \ldots, x_n$ where $x_{i+1} = x_i + \delta$, a typical **finite difference** approach is:

$$\left.\frac{\partial f(x)}{\partial x}\right|_{x=x_i} \approx \frac{f(x_{i+1}) - f(x_i)}{\delta}.$$

When the sample spacing is $\delta = 1$, this approximation simplifies to

$$f'(x_i) \triangleq \left.\frac{\partial f(x)}{\partial x}\right|_{x=x_i} \approx f(x_{i+1}) - f(x_i).$$

We can express this relation for all x_i samples via the matrix-vector product

$$\begin{bmatrix} f'(x_1) \\ f'(x_2) \\ \vdots \\ f'(x_n) \end{bmatrix} \approx \boldsymbol{D}_n \begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{bmatrix},$$

where $\boldsymbol{D}_n$ is the so-called first **finite difference** matrix defined by

$$\boldsymbol{D}_n = \begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \\ 1 & & & & -1 \end{bmatrix}.$$

Here we choose to set $\boldsymbol{D}_n[n, 1] = 1$, which corresponds to the (perhaps unexpected) approximation $f'(x_n) \approx f(x_1) - f(x_n)$. This choice is called a **periodic boundary condition** because essentially we are assuming that the domain wraps around. We make this assumption because the resulting $\boldsymbol{D}_n$ is a **circulant matrix** so its eigenvectors can be computed in closed form! (See Ch. 7.)
The goal of this problem is for you to derive and implement the analog of $\boldsymbol{D}_n$ for 2D differentiation. Let $f(x, y)$ be a function of two variables. We can approximate its partial derivatives using finite differences as follows:

$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x+1,y) - f(x,y)}{(x+1) - x} = f(x+1,y) - f(x,y) \tag{1}$$

$$\frac{\partial f(x,y)}{\partial y} \approx \frac{f(x,y+1) - f(x,y)}{(y+1) - y} = f(x,y+1) - f(x,y). \tag{2}$$

To simplify notation, define the $m \times n$ matrices `FXY`, `DFDX`, and `DFDY` having elements as follows:

$$\begin{aligned} \texttt{FXY}[i,j] &= f(i,j) \\ \texttt{DFDX}[i,j] &= \frac{\partial f(i,j)}{\partial x} \\ \texttt{DFDY}[i,j] &= \frac{\partial f(i,j)}{\partial y} \end{aligned}$$

The x coordinate is along the column of `FXY` and the y coordinate is along the row of `FXY`, so we can think `FXY[x,y]`.
Define corresponding vectors `fxy`, `dfdx`, and `dfdy` in $\mathbb{R}^{mn}$ to be vectorized versions[1] of `FXY`, `DFDX`, and `DFDY`.
With this notation, we can succinctly express equations (1) and (2) as

$$\begin{bmatrix} \texttt{dfdx} \\ \texttt{dfdy} \end{bmatrix} = \boldsymbol{A}\ \texttt{fxy},$$

where $\boldsymbol{A}$ is a $2mn \times mn$ matrix.

[1] In `Julia`, `fxy = FXY[:]` and `fxy = vec(FXY)`.

To help explain the notation, here is a concrete example:

$$\texttt{FXY} = \begin{bmatrix} 0 & 1 & 3 \\ 4 & 9 & 15 \end{bmatrix} \Rightarrow \texttt{DFDX} = \begin{bmatrix} 4 & 8 & 12 \\ -4 & -8 & -12 \end{bmatrix}, \quad \texttt{DFDY} = \begin{bmatrix} 1 & 2 & -3 \\ 5 & 6 & -11 \end{bmatrix}, \quad \texttt{fxy} = \begin{bmatrix} 0 \\ 4 \\ 1 \\ 9 \\ 3 \\ 15 \end{bmatrix}, \quad \texttt{dfdx} = \begin{bmatrix} 4 \\ -4 \\ 8 \\ -8 \\ 12 \\ -12 \end{bmatrix}.$$

(a) Find an expression for $\boldsymbol{A}$ in terms of the first difference matrices $\boldsymbol{D}_n$, $\boldsymbol{D}_m$, appropriately sized identity matrices, and appropriate **Kronecker products** of these matrices. Use periodic boundary conditions.

Hint: Start with $m = n = 3$. Look for ways to use Kronecker product(s) and its "vec trick."

Hint: To make a 5×5 identity matrix use:

```
using LinearAlgebra: I
I(5)
```

(b) Once you have determined $\boldsymbol{A}$, write the function `first_diffs_2d_matrix` that takes as input the dimensions m and n of `FXY` and returns the appropriate $\boldsymbol{A}$ matrix, stored in **sparse** format.

In `Julia`, your file should be named `first_diffs_2d_matrix.jl` and should contain the following function:

```
"""
    A = first_diffs_2d_matrix(m, n)

In:
- `m` and `n` are positive integers

Out:
- `A` is a `2mn × mn` sparse matrix such that `A * vec(X)` computes the
first differences down the columns (along x direction)
and across the (along y direction) of the `m × n` matrix `X`.
"""
function first_diffs_2d_matrix(m, n)
```

Email your solution as an attachment to `eecs551@autograder.eecs.umich.edu`.

Hint. Be sure to have `using [packagename]` at the top of your code for any package you use.
The matrix $\boldsymbol{A}$ can be gigantic. For example, suppose $m = 550$ and $n = 430$, then $\boldsymbol{A}$ is a $473{,}000 \times 236{,}500$ matrix that would require 833 GB of RAM if stored as a full double precision matrix! However, $\boldsymbol{A}$ has only $4mn$ non-zero entries, so it is very **sparse**.
Matrices with many zero-valued elements are quite common in applications. For normal arrays, `Julia` (and other languages) stores zeros in the same way it stores other numeric values, so having many zero elements can use memory space unnecessarily and can sometimes require extra computing time.
Sparse matrices provide an efficient way to store data that has a large percentage of zero elements. While full matrices internally store every element in memory regardless of value, a **sparse matrix** data structure stores only the nonzero elements and their row indices. Using sparse matrices can significantly reduce the amount of memory required for data storage.
In `Julia`, to create a sparse matrix somewhat similar to $\boldsymbol{D}_n$ above one can use either the `spdiagm` command:

```
using SparseArrays
n = 5
A = spdiagm(0 => 1:n, -1 => ones(n-1))
```

or a loop:

```
n = 5
A = spzeros(n,n)
for i=1:n-1
    A[i,i], A[i,i+1] = i, 1
end
```

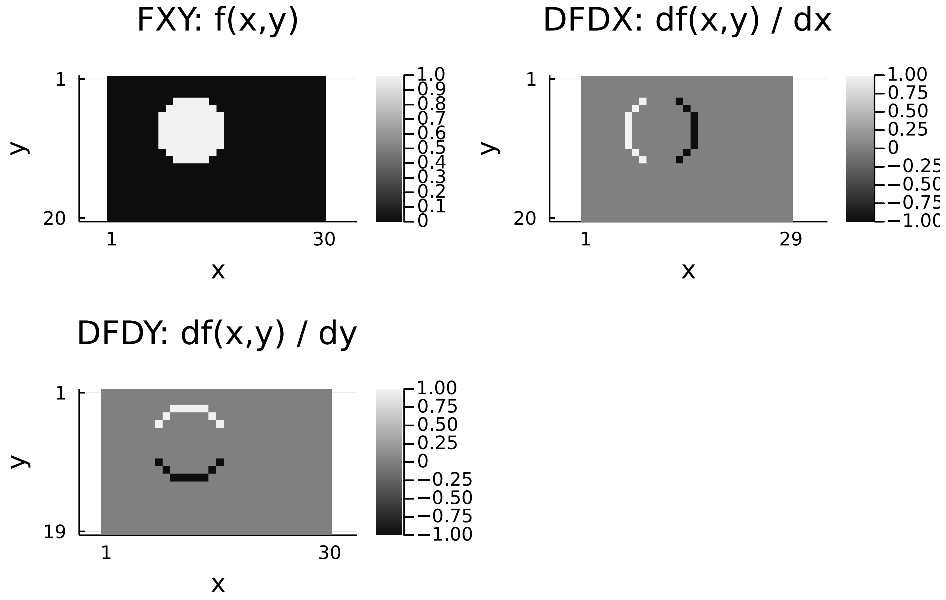
You should try one or both of these out and modify one of them to prepare your answer. For details, see the SparseArrays documentation.

For any Julia assignment, you should always try to think of your own ways of testing your function before submitting to the auto-grader. You do get unlimited tries with the auto-grader, but using the feedback from the auto-grader is a poor way to debug. By designing your own tests, you can examine their output interactively and fix bugs more intelligently.

In this problem your function is designed to compute finite difference approximations to derivatives along x and y. If you create a $m \times n$ array `X` that is, say, a picture of a disk, then the finite derivatives will be mostly zero except near the edges of the disk. (This property is related to an image processing application called edge detection that is discussed in EECS 556.)

Here is Julia code for making a (digital/sampled) disk and showing pictures of FXY, DFDX, DFDY.

```julia
using MIRTjim: jim
using Plots: savefig
m = 30; n = 20; X = Float64.([(x-12)^2+(y-8)^2 < 5^2 for x=1:m, y=1:n])
dfdx = diff(X; dims=1)
dfdy = diff(X; dims=2)
jim(
    jim(X, "FXY: f(x,y)"; xlabel="x", ylabel="y"),
    jim(dfdx, "DFDX: df(x,y) / dx"; xlabel="x", ylabel="y"),
    jim(dfdy, "DFDY: df(x,y) / dy"; xlabel="x", ylabel="y"),
) # savefig("hp074_disk.pdf")
```



You should make similar examples to test your function.

Pr. 11.
(Hand-written digit classification using inner products: discussion task)

This task illustrates how to use inner products for classifying handwritten digits. We focus on just the two digits “0” and “1”, although the principles generalize to all digits.

Download the `task-1-classify-angle.ipynb` jupyter notebook file from Canvas and follow all instructions to complete the task. You may work individually, but we recommend that you work in pairs or groups of three.

When you are finished, upload your solutions to gradescope. Note that the submission for the task is separate from the rest of HW02 because the task allows you to submit as a group. Only upload one submission per group! Whoever uploads the group submission should add all group members in gradescope, using the “View or edit group” option on the right-hand sidebar after uploading a PDF and matching pages. Make sure to add all group members, because this is how they will receive credit!

Non-graded problem(s) below
(Solutions will be provided for self check; do not submit to gradescope.)

Pr. 12.
Let $\boldsymbol{B} = \boldsymbol{T}^{-1}\boldsymbol{A}\boldsymbol{T}$ for an invertible matrix $\boldsymbol{T}$. Determine the relationship between eigenvalues and eigenvectors of $\boldsymbol{B}$ and the eigenvalues and eigenvectors of $\boldsymbol{A}$. Explain. The matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, when thus related, are called **similar**. (Use the convention throughout that eigenvectors are normalized to have unit norm.)

Pr. 13.
Let $\boldsymbol{X} \in \mathbb{F}^{M\times N}$. Using an **SVD** only, show that if $\boldsymbol{X}'\boldsymbol{X} = \boldsymbol{0}$, then $\boldsymbol{X} = \boldsymbol{0}$.

Pr. 14.
Let $\boldsymbol{A} \in \mathbb{C}^{N\times N} = \boldsymbol{U\Sigma V}'$ have **rank** N, *i.e.*, N nonzero singular values.

(a) Express $\boldsymbol{A}^{-1}$ in terms of the **SVD** of $\boldsymbol{A}$ without using any inverse $(\ )^{-1}$.

(b) Determine an SVD for $\boldsymbol{A}^{-1}$.

Pr. 15.
Let $\boldsymbol{A} = \boldsymbol{U\Sigma V}'$ denote an **SVD** of $M \times N$ matrix $\boldsymbol{A}$ having rank r. We can write $\boldsymbol{A}$ in terms of its first r singular values and singular vectors as follows

$$\boldsymbol{A} = \sum_{i=1}^{r} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i',$$

where $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ denote the ith columns of $\boldsymbol{U}$ and $\boldsymbol{V}$ respectively, and σ_i is the (i,i) entry of $\boldsymbol{\Sigma}$.
Recall that the squared **Frobenius norm** of $\boldsymbol{A}$ is $\|\boldsymbol{A}\|_F = \sqrt{\text{trace}(\boldsymbol{A}'\boldsymbol{A})} = \sqrt{\text{trace}(\boldsymbol{A}\boldsymbol{A}')}$.
This problem explores some simple but important properties of the **singular vectors** of $\boldsymbol{A}$. For each case, write an answer in terms of the SVD components like $\boldsymbol{U}$, $\boldsymbol{\Sigma}$, $\boldsymbol{V}$, $\boldsymbol{u}_i$, $\boldsymbol{v}_i$, σ_i, etc.

(a) (1) $\boldsymbol{A}\boldsymbol{v}_i = ?$
(2) $(\boldsymbol{A}\boldsymbol{v}_i)'(\boldsymbol{A}\boldsymbol{v}_j) = ?$
(3) $\boldsymbol{A}'\boldsymbol{u}_i = ?$ when $i \in \{1,\ldots,r\}$?
What if $i \in \{r+1,\ldots,M\}$?
(4) $(\boldsymbol{A}'\boldsymbol{u}_i)'(\boldsymbol{A}'\boldsymbol{u}_j) = ?$
(5) $\|\boldsymbol{A}\boldsymbol{v}_i\|_2^2 = ?$

(b) (1) $\boldsymbol{A}\boldsymbol{v}_i\boldsymbol{v}_i' = ?$
(2) $\boldsymbol{A}'\boldsymbol{u}_i\boldsymbol{u}_i' = ?$ when $i \in \{1,\ldots,r\}$?
What if $i \in \{r+1,\ldots,M\}$?
(3) $\|\boldsymbol{A}\boldsymbol{v}_i\boldsymbol{v}_i'\|_F = ?$
(4) $\|\boldsymbol{A}'\boldsymbol{u}_i\boldsymbol{u}_i'\|_F = ?$
(5) $\|\boldsymbol{A}\boldsymbol{v}_i\boldsymbol{u}_j'\|_F = ?$
(6) $\|\boldsymbol{A}\boldsymbol{v}_i\boldsymbol{v}_j'\|_F = ?$
(7) $\boldsymbol{A}\boldsymbol{A}' = ?$
(8) $\boldsymbol{A}'\boldsymbol{A} = ?$
(9) $\|\boldsymbol{A}'\boldsymbol{A}\|_F = ?$
(10) $\|\boldsymbol{A}\boldsymbol{A}'\|_F = ?$

(c) (1) For $1 \le k \le r$, $\boldsymbol{U}[:,1{:}k]'\boldsymbol{A} = ?$
(2) For $1 \le k \le r$, $\boldsymbol{A}\boldsymbol{V}[:,1{:}k] = ?$
(3) For $1 \le k \le M$, $\boldsymbol{U}[:,1{:}k]\boldsymbol{U}[:,1{:}k]'\boldsymbol{A} = ?$
(4) For $1 \le k \le N$, $\boldsymbol{A}\boldsymbol{V}[:,1{:}k]\boldsymbol{V}[:,1{:}k]' = ?$
(5) For $1 \le k \le r$, $\boldsymbol{U}[:,1{:}k]'\boldsymbol{A}\boldsymbol{V}[:,1{:}k] = ?$

Pr. 16.

After defining matrix-matrix multiplication in terms of the elements of those matrices, the course notes describe 4 different versions. A F19 student suggested in class that there is another version. In fact there are two more versions. Study Versions 1-4 and describe two more versions that are distinct from the definition and the given versions.
Just describe them mathematically; code is not required.
Discuss any possible advantages or disadvantages of these versions.