How to repair \texttt{project\_onto}?

Don’t change the procedure. Fix the spec.

Require that \texttt{vlist} consists of \textbf{mutually orthogonal} vectors:

the \textit{i}th vector in the list is orthogonal to the \textit{j}th vector in the list for every \( i \neq j \).
The return of `project_onto`

- **input:** a vector $\mathbf{b}$, a list `vlist` $[\mathbf{v}_1, \ldots, \mathbf{v}_n]$ of mutually orthogonal vectors
- **output:** the projection of $\mathbf{b}$ onto the space spanned by $\mathbf{v}_1, \ldots, \mathbf{v}_n$

```python
def project_onto(b, vlist):
    return sum([project_along(b, v) for v in vlist])
```

Let $\hat{\mathbf{b}}$ be the result.

Need to prove

- $\hat{\mathbf{b}}$ lies in $\text{Span} \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$, and
- $\mathbf{b} - \hat{\mathbf{b}}$ is orthogonal to $\text{Span} \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ Suffices to show that $\mathbf{b} - \hat{\mathbf{b}}$ is orthogonal to each of $\mathbf{v}_1, \ldots, \mathbf{v}_n$ for then it is orthogonal to every linear combination
Proving the correctness of `project_onto`

def project_onto(b, vlist): return sum([project_along(b, v) for v in vlist])

Let \( \hat{b} \) be the result.

Need to prove

1. \( \hat{b} \) lies in \( \text{Span} \{\mathbf{v}_1, \ldots, \mathbf{v}_n\} \), and

2. \( \mathbf{b} - \hat{b} \) is orthogonal to \( \text{Span} \{\mathbf{v}_1, \ldots, \mathbf{v}_n\} \) Suffices to show that \( \mathbf{b} - \hat{b} \) is orthogonal to each of \( \mathbf{v}_1, \ldots, \mathbf{v}_n \) for then it is orthogonal to every linear combination

(1) By correctness of `project_along(b, v)`, the result is a scalar multiple of \( v \) for each vector \( v \) in \( vlist \). Thus \( \hat{b} = \sigma_1 \mathbf{v}_1 + \ldots \sigma_n \mathbf{v}_n \) where \( \sigma_1, \ldots, \sigma_n \) are the scalars.

This is a linear combination of \( \mathbf{v}_1, \ldots, \mathbf{v}_n \), so \( \hat{b} \) belongs to \( \text{Span} \{\mathbf{v}_1, \ldots, \mathbf{v}_n\} \).
Proving the correctness of \texttt{project\_onto} \\

Need to prove  
1. $\hat{\mathbf{b}}$ lies in Span $\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$, and  
2. $\mathbf{b} - \hat{\mathbf{b}}$ is orthogonal to Span $\{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$  

Suffices to show that $\mathbf{b} - \hat{\mathbf{b}}$ is orthogonal to each of $\mathbf{v}_1, \ldots, \mathbf{v}_n$ for then it is orthogonal to every linear combination  

(2) For $i = 1, 2, \ldots, n,$  

\[
\begin{align*}
\langle \mathbf{b} - \hat{\mathbf{b}}, \mathbf{v}_i \rangle &= \langle \mathbf{b}, \mathbf{v}_i \rangle - \langle \hat{\mathbf{b}}, \mathbf{v}_i \rangle \\
&= \langle \mathbf{b}, \mathbf{v}_i \rangle - \langle \sigma_1 \mathbf{v}_1 - \sigma_2 \mathbf{v}_2 + \cdots - \sigma_i \mathbf{v}_i - \cdots - \sigma_n \mathbf{v}_n, \mathbf{v}_i \rangle \\
&= \langle \mathbf{b}, \mathbf{v}_i \rangle - \sigma_1 \langle \mathbf{v}_1, \mathbf{v}_i \rangle - \sigma_2 \langle \mathbf{v}_2, \mathbf{v}_i \rangle - \cdots - \sigma_i \langle \mathbf{v}_i, \mathbf{v}_i \rangle - \cdots - \sigma_n \langle \mathbf{v}_n, \mathbf{v}_i \rangle \\
&= \langle \mathbf{b}, \mathbf{v}_i \rangle - 0 - 0 - \cdots - \sigma_i \langle \mathbf{v}_i, \mathbf{v}_i \rangle - \cdots - 0 \\
&= \langle \mathbf{b}, \mathbf{v}_i \rangle - \sigma_i \langle \mathbf{v}_i, \mathbf{v}_i \rangle \\
&= \langle \mathbf{b} \| \mathbf{v}_i + \mathbf{b}_i^\perp, \mathbf{v}_i \rangle - \sigma_i \langle \mathbf{v}_i, \mathbf{v}_i \rangle \\
&= \langle \mathbf{b} \| \mathbf{v}_i, \mathbf{v}_i \rangle + \langle \mathbf{b}_i^\perp \mathbf{v}_i, \mathbf{v}_i \rangle - \sigma_i \langle \mathbf{v}_i, \mathbf{v}_i \rangle \\
&= \langle \sigma_i \mathbf{v}_i, \mathbf{v}_i \rangle + 0 - \sigma_i \langle \mathbf{v}_i, \mathbf{v}_i \rangle = 0
\end{align*}
\]
A new subroutine: `project_orthogonal(b, vlist)`

We have proved that `project_onto(b, vlist)` satisfies its spec:

- **input**: vector `b`, list `vlist` of mutually orthogonal vectors
- **output**: projection of `b` onto the span of vectors in `vlist`

Use this to build a subroutine `project_orthogonal(b, vlist)` with spec:

- **input**: vector `b`, list `vlist` of mutually orthogonal vectors
- **output**: projection of `b` orthogonal to the span of vectors in `vlist`

```python
def project_orthogonal(b, vlist): return b - project_onto(b, vlist)
```
Building an orthogonal set of generators

**Original stated goal:**
Find the projection of $\mathbf{b}$ onto the space $\mathcal{V}$ spanned by arbitrary vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$.

So far we know how to find the projection of $\mathbf{b}$ onto the space spanned by mutually orthogonal vectors.

This would suffice if we had a procedure that, given arbitrary vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$, computed mutually orthogonal vectors $\mathbf{v}^*_1, \ldots, \mathbf{v}^*_n$ that span the same space.

We consider a new problem: **orthogonalization**:

- **Input:** A list $[\mathbf{v}_1, \ldots, \mathbf{v}_n]$ of vectors over the reals
- **Output:** A list of mutually orthogonal vectors $\mathbf{v}^*_1, \ldots, \mathbf{v}^*_n$ such that

$$\text{Span} \ \{\mathbf{v}^*_1, \ldots, \mathbf{v}^*_n\} = \text{Span} \ \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$$

How can we solve this problem?
The orthogonalize procedure

**Idea:** Use `project_orthogonal` iteratively to make a longer and longer list of mutually orthogonal vectors.

- First consider $\mathbf{v}_1$. Define $\mathbf{v}_1^* := \mathbf{v}_1$ since the set $\{\mathbf{v}_1^*\}$ is trivially a set of mutually orthogonal vectors.
- Next, define $\mathbf{v}_2^*$ to be the projection of $\mathbf{v}_2$ orthogonal to $\mathbf{v}_1^*$. 
- Now $\{\mathbf{v}_1^*, \mathbf{v}_2^*\}$ is a set of mutually orthogonal vectors.
- Next, define $\mathbf{v}_3^*$ to be the projection of $\mathbf{v}_3$ orthogonal to $\mathbf{v}_1^*$ and $\mathbf{v}_2^*$, so $\{\mathbf{v}_1^*, \mathbf{v}_2^*, \mathbf{v}_3^*\}$ is a set of mutually orthogonal vectors.

In each step, we use `project_orthogonal` to find the next orthogonal vector.

In the $i^{th}$ iteration, we project $\mathbf{v}_i$ orthogonal to $\mathbf{v}_1^*, \ldots, \mathbf{v}_{i-1}^*$ to find $\mathbf{v}_i^*$.

```python
def orthogonalize(vlist):
    vstarlist = []
    for v in vlist:
        vstarlist.append(project_orthogonal(v, vstarlist))
    return vstarlist
```
def orthogonalize(vlist):
    vstarlist = []
    for v in vlist:
        vstarlist.append(project_orthogonal(v, vstarlist))
    return vstarlist

Lemma: Throughout the execution of orthogonalize, the vectors in vstarlist are mutually orthogonal.

In particular, the list vstarlist at the end of the execution, which is the list returned, consists of mutually orthogonal vectors.

Proof: by induction, using the fact that each vector added to vstarlist is orthogonal to all the vectors already in the list.
Example of orthogonalize

**Example:** When `orthogonalize` is called on a `vlist` consisting of vectors

\[
\mathbf{v}_1 = [2, 0, 0], \mathbf{v}_2 = [1, 2, 2], \mathbf{v}_3 = [1, 0, 2]
\]

it returns the list `vstarlist` consisting of

\[
\mathbf{v}_1^* = [2, 0, 0], \mathbf{v}_2^* = [0, 2, 2], \mathbf{v}_3^* = [0, -1, 1]
\]

(1) In the first iteration, when \( \mathbf{v} \) is \( \mathbf{v}_1 \), `vstarlist` is empty, so the first vector \( \mathbf{v}_1^* \) added to `vstarlist` is \( \mathbf{v}_1 \) itself.

(2) In the second iteration, when \( \mathbf{v} \) is \( \mathbf{v}_2 \), `vstarlist` consists only of \( \mathbf{v}_1^* \). The projection of \( \mathbf{v}_2 \) orthogonal to \( \mathbf{v}_1^* \) is

\[
\mathbf{v}_2 - \frac{\langle \mathbf{v}_2, \mathbf{v}_1^* \rangle}{\langle \mathbf{v}_1^*, \mathbf{v}_1^* \rangle} \mathbf{v}_1^* = [1, 2, 2] - \frac{2}{4} [2, 0, 0] = [0, 2, 2]
\]

so \( \mathbf{v}_2^* = [0, 2, 2] \) is added to `vstarlist`.

(3) In the third iteration, when \( \mathbf{v} \) is \( \mathbf{v}_3 \), `vstarlist` consists of \( \mathbf{v}_1^* \) and \( \mathbf{v}_2^* \). The projection of \( \mathbf{v}_3 \) orthogonal to \( \mathbf{v}_1^* \) is \( [0, 0, 2] \), and the projection of \( [0, 0, 2] \) orthogonal to \( \mathbf{v}_2^* \) is

\[
[0, 0, 2] - \frac{1}{2} [0, 2, 2] = [0, -1, 1]
\]

so \( \mathbf{v}_3^* = [0, -1, 1] \) is added to `vstarlist`
Correctness of the orthogonalize procedure, Part II

**Lemma:** Consider orthogonalize applied to an $n$-element list $[\mathbf{v}_1, \ldots, \mathbf{v}_n]$. After $i$ iterations of the algorithm, $\text{Span } v_{\text{star list}} = \text{Span } \{\mathbf{v}_1, \ldots, \mathbf{v}_i\}$.

**Proof:** by induction on $i$.

The case $i = 0$ is trivial.

After $i - 1$ iterations, $v_{\text{star list}}$ consists of vectors $\mathbf{v}^*_1, \ldots, \mathbf{v}^*_{i-1}$.

Assume the lemma holds at this point. This means that

$$\text{Span } \{\mathbf{v}^*_1, \ldots, \mathbf{v}^*_{i-1}\} = \text{Span } \{\mathbf{v}_1, \ldots, \mathbf{v}_{i-1}\}$$

By adding the vector $\mathbf{v}_i$ to sets on both sides, we obtain

$$\text{Span } \{\mathbf{v}^*_1, \ldots, \mathbf{v}^*_{i-1}, \mathbf{v}_i\} = \text{Span } \{\mathbf{v}_1, \ldots, \mathbf{v}_{i-1}, \mathbf{v}_i\}$$

... It therefore remains only to show that $\text{Span } \{\mathbf{v}^*_1, \ldots, \mathbf{v}^*_{i-1}, \mathbf{v}_i\} = \text{Span } \{\mathbf{v}^*_1, \ldots, \mathbf{v}^*_{i-1}, \mathbf{v}_i\}$.

The $i^{th}$ iteration computes $\mathbf{v}^*_i$ using \texttt{project_orthogonal}(\mathbf{v}_i, [\mathbf{v}^*_1, \ldots, \mathbf{v}^*_{i-1}]).

There are scalars $\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{i,i-1}$ such that

$$\mathbf{v}_i = \alpha_{i1}\mathbf{v}_1^* + \cdots + \alpha_{i,i-1}\mathbf{v}_{i-1}^* + \mathbf{v}_i^*$$

This equation shows that any linear combination of $\mathbf{v}^*_1, \ldots, \mathbf{v}^*_{i-1}, \mathbf{v}_i$ can be transformed into a linear combination of $\mathbf{v}^*_1, \ldots, \mathbf{v}^*_{i-1}, \mathbf{v}_i$ and vice versa.

QED
Correctness of the orthogonalize procedure, Part II

**Lemma:** Consider \texttt{orthogonalize} applied to an $n$-element list $[\mathbf{v}_1, \ldots, \mathbf{v}_n]$. After $i$ iterations of the algorithm, $\text{Span } \mathbf{v}_{\text{star list}} = \text{Span } \{\mathbf{v}_1, \ldots, \mathbf{v}_i\}$.

**Proof:** by induction on $i$.

... It therefore remains only to show that $\text{Span } \{\mathbf{v}_1^*, \ldots, \mathbf{v}_{i-1}^*, \mathbf{v}_i^*\} = \text{Span } \{\mathbf{v}_1^*, \ldots, \mathbf{v}_{i-1}^*, \mathbf{v}_i\}$.

The $i^{th}$ iteration computes $\mathbf{v}_i^*$ using \texttt{project\_orthogonal}(\mathbf{v}_i, [\mathbf{v}_1^*, \ldots, \mathbf{v}_{i-1}^*]).

There are scalars $\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{i,i-1}$ such that

$$
\mathbf{v}_i = \alpha_{i1}\mathbf{v}_1^* + \cdots + \alpha_{i-1,i}\mathbf{v}_{i-1}^* + \mathbf{v}_i^*
$$

This equation shows that any linear combination of $\mathbf{v}_1^*, \mathbf{v}_2^*, \ldots, \mathbf{v}_{i-1}^*, \mathbf{v}_i$ can be transformed into a linear combination of $\mathbf{v}_1^*, \mathbf{v}_2^*, \ldots, \mathbf{v}_{i-1}^*, \mathbf{v}_i^*$ and vice versa. \(\text{QED}\)
Order in orthogonalize

Order matters!

Suppose you run the procedure `orthogonalize` twice, once with a list of vectors and once with the reverse of that list.

The output lists will **not** be the reverses of each other.

Contrast with `project_orthogonal(b, vlist)`.`

The projection of a vector `b` orthogonal to a vector space is unique, so in principle the order of vectors in `vlist` doesn’t affect the output of `project_orthogonal(b, vlist)`.
Matrix form for orthogonalize

For `project_orthogonal`, we had

\[
\begin{bmatrix}
  b \\
  v_0 & \cdots & v_n \\
  v_n^\top
\end{bmatrix}
= \begin{bmatrix}
  \alpha_0 \\
  \vdots \\
  \alpha_n \\
  1
\end{bmatrix}
\]

For `orthogonalize`, we have

\[
\begin{bmatrix}
  v_0 \\
  v_1 \\
  v_2 \\
  v_3
\end{bmatrix}
= \begin{bmatrix}
  v_0^* \\
  v_0^* \\
  v_0^* \\
  v_0^*
\end{bmatrix}
\begin{bmatrix}
  1 \\
  \alpha_{01} \\
  \alpha_{02} \\
  \alpha_{03}
\end{bmatrix}
\]

\[
\begin{bmatrix}
  v_0 \\
  v_1 \\
  v_2 \\
  v_3
\end{bmatrix}
= \begin{bmatrix}
  v_0^* \\
  v_0^* \\
  v_0^* \\
  v_0^*
\end{bmatrix}
\begin{bmatrix}
  1 \\
  1 \\
  \alpha_{12} \\
  \alpha_{23}
\end{bmatrix}
\]

The two matrices on the right are special:

- Columns of first one are mutually orthogonal.
- Second is upper triangular.

We will use these properties in algorithms.
Example of matrix form for orthogonalize

Example: for vlist consisting of vectors

\[
\begin{bmatrix}
2 \\
0 \\
0
\end{bmatrix}, \begin{bmatrix}
1 \\
2 \\
2
\end{bmatrix}, \begin{bmatrix}
1 \\
0 \\
2
\end{bmatrix}
\]

we saw that the output list vstarlist of orthogonal vectors consists of

\[
\begin{bmatrix}
2 \\
0 \\
0
\end{bmatrix}, \begin{bmatrix}
0 \\
2 \\
2
\end{bmatrix}, \begin{bmatrix}
0 \\
-1 \\
1
\end{bmatrix}
\]

The corresponding matrix equation is

\[
\begin{bmatrix}
\begin{bmatrix}
2 \\
0 \\
0
\end{bmatrix} & \begin{bmatrix}
1 \\
0 \\
2
\end{bmatrix} & \begin{bmatrix}
1 \\
0 \\
2
\end{bmatrix}
\end{bmatrix} = \begin{bmatrix}
2 & 0 & 0 \\
0 & 2 & -1 \\
0 & 2 & 1
\end{bmatrix} \begin{bmatrix}
1 & 0.5 & 0.5 \\
1 & 0.5 & 1
\end{bmatrix}
\]

Solving closest point in the span of many vectors

Let $\mathcal{V} = \text{Span}\ \{v_0, \ldots, v_n\}$.

The vector in $\mathcal{V}$ closest to $b$ is $b_{\parallel\mathcal{V}}$, which is $b - b_{\perp\mathcal{V}}$.

There are two equivalent ways to find $b_{\perp\mathcal{V}}$,

**One method:**

Step 1: Apply orthogonalize to $v_0, \ldots, v_n$, and obtain $v_0^*, \ldots, v_n^*$.  
(Now $\mathcal{V} = \text{Span}\ \{v_0^*, \ldots, v_n^*\}$)

Step 2: Call $\text{project\_orthogonal}(b, [v_0^*, \ldots, v_n^*])$ and obtain $b_{\perp}$ as the result.

**Another method:** Exactly the same computations take place when orthogonalize is applied to $[v_0, \ldots, v_n, b]$ to obtain $[v_0^*, \ldots, v_n^*, b^*]$.

In the last iteration of orthogonalize, the vector $b^*$ is obtained by projecting $b$ orthogonal to $v_0^*, \ldots, v_n^*$. Thus $b^* = b_{\perp}$. 
Solving other problems using orthogonalization

We’ve shown how orthogonalize can be used to find the vector in Span \( \{v_0, \ldots, v_n\} \) closest to \( b \), namely \( b^\parallel \).

Later we give an algorithm to find the coordinate representation of \( b^\parallel \) in terms of \( \{v_0, \ldots, v_n\} \).

First we will see how we can use orthogonalization to solve other computational problems.

We need to prove something about mutually orthogonal vectors....
Mutually orthogonal nonzero vectors are linearly independent

**Proposition:** Mutually orthogonal nonzero vectors are linearly independent.

**Proof:** Let \( \mathbf{v}_0^*, \mathbf{v}_1^*, \ldots, \mathbf{v}_n^* \) be mutually orthogonal nonzero vectors.

Suppose \( \alpha_0, \alpha_1, \ldots, \alpha_n \) are coefficients such that

\[
0 = \alpha_0 \mathbf{v}_0^* + \alpha_1 \mathbf{v}_1^* + \cdots + \alpha_n \mathbf{v}_n^*
\]

We must show that therefore the coefficients are all zero.

To show that \( \alpha_0 \) is zero, take inner product with \( \mathbf{v}_0^* \) on both sides:

\[
\langle \mathbf{v}_0^*, 0 \rangle = \langle \mathbf{v}_0^*, \alpha_0 \mathbf{v}_0^* + \alpha_1 \mathbf{v}_1^* + \cdots + \alpha_n \mathbf{v}_n^* \rangle
= \alpha_0 \langle \mathbf{v}_0^*, \mathbf{v}_0^* \rangle + \alpha_1 \langle \mathbf{v}_0^*, \mathbf{v}_1^* \rangle + \cdots + \alpha_n \langle \mathbf{v}_0^*, \mathbf{v}_n^* \rangle
= \alpha_0 \| \mathbf{v}_0^* \|^2 + \alpha_1 0 + \cdots + \alpha_n 0
= \alpha_0 \| \mathbf{v}_0^* \|^2
\]

The inner product \( \langle \mathbf{v}_0^*, 0 \rangle \) is zero, so \( \alpha_0 \| \mathbf{v}_0^* \|^2 = 0 \). Since \( \mathbf{v}_0^* \) is nonzero, its norm is nonzero, so the only solution is \( \alpha_0 = 0 \).

Can similarly show that \( \alpha_1 = \cdots = \alpha_n = 0 \).

QED