Errata for Edition 1 of *Coding the Matrix*, April 1, 2015

Your copy might not contain some of these errors. They do not occur in the copies currently being sold as of the above date.

- Section 0.3: “... the input is a *pre-image* of the input” should be “... the input is a *pre-image* of the output”.

- Figure 4 in Section 0.3.8: The figure should be as follows:

  ![Figure 4](image)

- Definition 0.3.14: “there exists $x \in A$ such that $f(x) = z$” should be “there exists $x \in D$ such that $f(x) = z$.”

- Section 0.5.4: At the end of the section labeled *Mutating a set*,

  ```python
  >>> U=S.copy()
  >>> U.add(5)
  >>> S
  {1, 3}
  ```

  should end with

  ```python
  >>> S
  {6}
  ```

- Problem 0.8.5: “row(p)” should be “row(p, n)”.

- Section 1.4.1: “Using the fact that $i^2 = 1$” should be “Using the fact that $i^2 = -1$”

- Section 1.4.5: The diagram illustrating rotation by 90 degrees is incorrect. The dots should form vertical lines to the left of the y-axis.

- Task 1.4.8 and 1.4.9: The figures accompanying these tasks are incorrect; they involve rotation by -90 degrees (i.e. 90 degrees clockwise) instead of 90 degrees (i.e. 90 degrees counterclockwise).

- Task 1.4.10: `image.file2image(filename)` returns a representation of a color image, namely a list of lists of 3-tuples. For the purpose of this task, you must transform it to a representation of a grayscale image, using `image.color2gray(·)`. Also, the pixel intensities are numbers between 0 and 255, not between 0 and 1. In this task, you should assign to `pts` the list of complex numbers $x + iy$ such that the image intensity of pixel $(x, y)$ is less than 120.

- Task 1.4.11: The task mentions `pts` but `S` is intended.

- Section 2.4.1: “or from $[-4, 4]$ to $[-3, -2]$” should be “or from $[-4, -4]$ to $[-3, -2]$”.

- Example 2.9.1: “Consider the dot-product of $[1, 1, 1, 1, 1]$ with $[10, 20, 0, 40, 100]$” should be “Consider the dot-product of $[1, 1, 1, 1, 1]$ with $[10, 20, 0, 40, -100]$.”
• Example 2.9.5:

\[ \text{cost} = \text{Vec}(D, \{ \text{hops : } $2.50/\text{ounce}, \text{malt : } $1.50/\text{pound}, \text{water : } $0.006, \text{yeast : } $0.45/\text{gram} \}) \]

should be

\[ \text{cost} = \text{Vec}(D, \{ \text{hops : } $2.50/\text{ounce}, \text{malt : } $1.50/\text{pound}, \text{water : } $0.006, \text{yeast : } $0.45/\text{gram} \}) \]

• Definition 2.9.6: “A linear equation is an equation of the form \( \mathbf{a} \cdot \mathbf{x} = \beta \), where ... is a vector variable.” should be “A linear equation is an equation of the form \( \mathbf{a} \cdot \mathbf{x} = \beta \), where ... \( \mathbf{x} \) is a vector variable.”

• Example 2.9.7: The total energy is not 625J but is 0.0845J, as the Python shows.

• Definition 2.9.10: “In general, a system of linear equations (often abbreviated linear system) is a collection of equations:

\[ \begin{align*}
\mathbf{a}_1 \cdot \mathbf{x} &= \beta_1 \\
\mathbf{a}_2 \cdot \mathbf{x} &= \beta_2 \\
& \vdots \\
\mathbf{a}_m \cdot \mathbf{x} &= \beta_m
\end{align*} \]

where \( \mathbf{x} \) is a vector variable. A solution is a vector \( \mathbf{x} \) that satisfies all the equations.” should be

“In general, a system of linear equations (often abbreviated linear system) is a collection of equations:

\[ \begin{align*}
\mathbf{a}_1 \cdot \mathbf{x} &= \beta_1 \\
\mathbf{a}_2 \cdot \mathbf{x} &= \beta_2 \\
& \vdots \\
\mathbf{a}_m \cdot \mathbf{x} &= \beta_m
\end{align*} \]

where \( \mathbf{x} \) is a vector variable. A solution is a vector \( \mathbf{x} \) that satisfies all the equations.”

• Quiz 2.9.13: The solution should be “The dot-products are \([2, 2, 0, 0]\).”

• Example 2.9.17:

  – “The password is \( \hat{x} = 10111 \)” should be “The password is \( \hat{x} = 10111 \),”
  – “Harry computes the dot-product \( \mathbf{a}_1 \cdot \hat{x} \)” should be “Harry computes the dot-product \( \mathbf{a}_1 \cdot \hat{x} \)”
  – “Harry computes the dot-product \( \mathbf{a}_2 \cdot \hat{x} \)” should be “Harry computes the dot-product \( \mathbf{a}_2 \cdot \hat{x} \)”
  – “Carole lets Harry log in if \( \beta_1 = \mathbf{a}_1 \cdot \hat{x}, \beta_2 = \mathbf{a}_2 \cdot \hat{x}, \ldots, \beta_k = \mathbf{a}_k \cdot \hat{x} \)” should be “Carole lets Harry log in if \( \beta_1 = \mathbf{a}_1 \cdot \hat{x}, \beta_2 = \mathbf{a}_2 \cdot \hat{x}, \ldots, \beta_k = \mathbf{a}_k \cdot \hat{x} \).”

• Example 2.9.28: “Eve can use the distributive property to compute the dot-product of this sum with the password even though she does not know the password:

\[ (01011 + 11110) \cdot \hat{x} = 01011 \cdot \hat{x} + 11110 \cdot \hat{x} = 0 + 1 = 1 \]

should be
“Eve can use the distributive property to compute the dot-product of this sum with the password \( x \) even though she does not know the password:

\[
(01011 \cdot x + 11110 \cdot x) = 01011 \cdot x + 11110 \cdot x = 0 + 1 = 1
\]

• Task 2.12.8: “Did you get the same result as in Task ???” should be “Did you get the same result as in Task 2.12.7?”

• Quiz 3.1.7: the solution

```python
def lin_comb(vlist, clist):
    return sum([coeff*v for (coeff, v) in zip(clist, vlist)])
```

should be

```python
def lin_comb(vlist, clist):
    return sum([coeff*v for (c, v) in zip(clist, vlist)])
```

• Section 3.2.4: The representation of the old generator \([0, 0, 1]\) in terms of the new generators \([1, 0, 0]\), \([1, 1, 0]\), and \([1, 1, 1]\) should be

\[
[0, 0, 1] = 0[1, 0, 0] - 1[1, 1, 0] + 1[1, 1, 1]
\]

• In Example 3.2.7, “The secret password is a vector \( \hat{a} \) over \( GF(2) \)... the human must respond with the dot-product \( a \cdot \hat{x} \)” should be “The secret password is a vector \( \hat{x} \) over \( GF(2) \)... the human must respond with the dot-product \( a \cdot \hat{x} \)”.

• Example 3.3.10: “This line can be represented as Span \([1, -2, -2]\)” should be “This line can be represented as Span \([-1, -2, 2]\)”.

• In Example 3.5.1, “There is one plane through the points \( u_1 = [1, 0, 4.4], u_2 = [0, 1, 4], \) and \( u_3 = [0, 0, 3] \)” should be “There is one plane through the points \( u_1 = [1, 0, 4.4], u_2 = [0, 1, 4], \) and \( u_3 = [0, 0, 3] \)”.

• Section 4.1.4: The pretty-printed form of \( M \) should be

```python
>>> print(M)
    # ?
  a | 2 1 3
  b | 20 10 30
```

for some order of the columns.

• Quiz 4.3.1: The pretty-printed form of \( \text{mat2vec}(M) \) should be

```python
>>> print(mat2vec(M))
('a', '#') ('a', '?') ('a', '@') ('b', '#') ('b', '?') ('b', '@')
```

for some order of the columns.

• Quiz 4.4.2: The pretty-printed form of \( \text{transpose}(M) \) should be

```python
2 3 1 20 30 10
```
for some order of the rows.

- Example 4.6.6: The matrix-vector product should be \([1, \ldots, 0]\).

- Section 4.7.2: “Applying Lemma 4.7.4 with \(v = \mathbf{u}_1\) and \(z = \mathbf{u}_1 - \mathbf{u}_2\)” should be “Applying Lemma 4.7.4 with \(v = \mathbf{u}_2\) and \(z = \mathbf{u}_1 - \mathbf{u}_2\)”

- Section 4.7.4: “because it is the same as \(H \ast \mathbf{c}\), which she can compute” should be “because it is the same as \(H \ast \tilde{\mathbf{c}}\), which she can compute”

- Section 4.11.2: “and here is the same diagram with the walk 3 c 2 e 4 2 shown” should be “and here is the same diagram with the walk 3 c 2 e 4 2 shown”

- Example 4.11.9: \(g \circ f([x_1, x_2])\) should be \([x_1 + x_2, x_1 + 2x_2]\).

- Example 4.11.15: The last matrix (in the third row) should be \[
\begin{bmatrix}
7 & 19 \\
4 & 8
\end{bmatrix}
\]

- Example 4.13.15: \(xvec_1\) should be \(x_1\) and \(xvec_2\) should be \(x_2\).

- The description of Task 4.14.2 comes before the heading “Task 4.14.2”.

- Section 4.15 (Geometry Lab): \textit{position} is used synonymously with \textit{location}.

- Section 4.14.6: “Hint: this uses the special property of the order of \(H\)’s rows” should be “Hint: this uses the special property of the order of \(H\)’s columns.”

- Problem 4.17.10 is the same as Problem 4.17.5.

- Problem 4.17.18: “For this procedure, the only operation you are allowed to do on \(A\) is vector-matrix multiplication, using the * operator: \(v \ast A\)” should be “For this procedure, the only operation you are allowed to do on \(B\) is vector-matrix multiplication, using the * operator: \(v \ast B\)”

- Problem 4.17.21: \(xvec_2\) should be \(x_2\).

- Definition 4.6.9: “An \(n \times n\) upper-triangular matrix \(A\) is a matrix with the property that \(A_{ij} = 0\) for \(j > i\)” should be “for \(i > j\).”

- Section 5.3.1: The Grow algorithm should be:

```python
def Grow(V):
    B = \emptyset
    repeat while possible:
        find a vector v in V that is not in Span \(B\), and put it in \(B\).
```

- Example 5.3.2: “Finally, note that Span \(B = \mathbb{R}^2\) and that neither \(v_1\) nor \(v_2\) alone could generate \(\mathbb{R}^2\)” should be \(\mathbb{R}^3\).
• Section 5.4.3: “Let $D$ be the set of nodes, e.g. $D = \{\text{Pembroke, Athletic, Main, Keeney, Wriston}\}$” should be “$D = \{\text{Pembroke, Athletic, Bio-Med, Main, Keeney, Wriston, Gregorian}\}$”

• Section 5.9.1: “The first vector $a_1$ goes horizontally from the top-left corner of the whiteboard element to the top-right corner” should be “The first vector $a_1$ goes horizontally from the top-left corner of the top-left sensor element to the top-right corner” and “The second vector $a_2$ goes vertically from the top-left corner of whiteboard to the bottom-left corner” should be “The second vector $a_2$ goes vertically from the top-left corner of the top-left sensor element to the bottom-left corner.”

$$L = [[0,0,0],[1,0,0],[0,1,0],[1,1,0],[0,0,1],[1,0,1],[0,1,1],[1,1,1]]$$

should be

$$L = [[0,0,0],[1,0,0],[0,1,0],[1,1,0],[0,0,1],[1,0,1],[0,1,1],[1,1,1]]$$

• Section 5.9.1, diagram: The point in the bottom-left-back of the cube should be labeled $(0,1,1)$ but is labeled $(0,1,0)$.

• Section 5.9.5: In “For the third basis vector $a_2$...” and “Remember that $a_2$ points from the camera center to the top-left corner of the sensor array, so $a_2 = (-.5, -.5, 1)^T$, $a_2$ should be $a_3$, and $a_3 = [0,0,1]$. The third vector in $\mathbf{c}_b$ has an extra 0.

• “The third vector $c_3$ goes from the origin (the camera center) to the top-right corner of whiteboard.” should be “The third vector $c_3$ goes from the origin (the camera center) to the top-left corner of the whiteboard.”

• Section 5.12.1:

• Section 5.12.6: The vector $\begin{bmatrix} x_1 \\ xvec2 \\ 1 \end{bmatrix}$ should be $\begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$

• Section 5.12.6: After Task 5.12.2, “Let $[y_1, y_2, y_3] = Hx$” should be “Let $[y_1, y_2, y_3] =  \hat{H}x$”.

• Problem 5.14.18: “Write and test a procedure $\text{superset\_basis}(S, L)$” should be “Write and test a procedure $\text{superset\_basis}(T, L)$”.

• Lemma 6.2.13 (Superset-Basis Lemma) states

For any vector space $\mathcal{V}$ and any linearly independent set $A$ of vectors, $\mathcal{V}$ has a basis that contains all of $A$. 

But should state

For any vector space $\mathcal{V}$ and any linearly independent set $A$ of vectors belonging to $\mathcal{V}$, $\mathcal{V}$ has a basis that contains all of $A$.

• Example 6.3.3: $\mathcal{V}$ is defined to be the null space of $\begin{bmatrix} 0 & 1 & -1 & 0 \\ 1 & 0 & 0 & -1 \end{bmatrix}$ but should be defined to be the null space of $\begin{bmatrix} 0 & 1 & -2 & 0 \\ 1 & 0 & 0 & -1 \end{bmatrix}$.

• Problem 6.7.3: The output condition says

$$\text{Span } S = \text{Span } S \cup \{z_1, z_2, \ldots, z_i\} - \{w_1, w_2, \ldots, w_k\}$$

but should say
\(\text{Span } S = \text{Span } S \cup \{z_1, z_2, \ldots, z_i\} - \{w_1, w_2, \ldots, w_i\}\)

- Section 7.7.1: \(xvec_1\) and \(xvec_2\) should be \(x_1\) and \(x_2\)
- Section 7.8.3: “We can represent the factorization of 1176 by the list \([(2, 3), (5, 2)]\), indicating that 1176 is obtained by multiplying together three 2’s and two 5’s” should be “We can represent the factorization of 1176 by the list \([(2, 3), (3, 1), (7, 2)]\), indicating that 1176 is obtained by multiplying together three 2’s, one 3 and two 7’s”, and “1176 = \(2^3 \cdot 7^2\)” should be “1176 = \(2^3 \cdot 7^2\)”.
- Task 7.8.7: For \(x = 61\), the factored entry has \(2 \cdot 3 \cdot 7 \cdot 13\). This should be \(2 \cdot 3 \cdot 7 \cdot 13\).
- Task 7.8.9: “\(gcd(a, b)\)” should be “\(gcd(a - b, N)\)”.
- Section 9.2: In new spec for \(\text{project\_orthogonal}(b, \text{vlist})\), output should be “the projection \(b^\perp\) of \(b\) orthogonal to the vectors in \(\text{vlist}\)”.
- Section 9.6.6: “These vectors span the same space as input vectors \(u_1, \ldots, u_k, w_1, \ldots, w^*_n\)” The * in \(w^*_n\) should not be there.
- Section 9.6.6: In the pseudocode for \(\text{find\_orthogonal\_complement}\), the last line should be return \([\text{w}^*_i\text{ for } i\text{ in } \{1, \ldots, n\}\text{ if } \text{w}^*_i\text{ is not the zero vector}]\)
- Task 10.9.16: The procedure \texttt{image\_round} should also ensure the numbers are between 0 and 255.
- Proof of Lemma 11.3.6: “Let \(V^*\) be the space dual to \(V\)” should be “Let \(V^*\) be the annihilator of \(V\)”, and “the dual of the dual” should be “the annihilator of the annihilator”.
- Section 11.3.3: “...we provide a module \texttt{svd} with a procedure \texttt{factor}(A) that, given a Mat \(A\), returns a triple \((U, \Sigma, V)\) such that \(A = U * \Sigma * V.\text{transpose}\)” should end “such that \(A = U * \Sigma * V.\text{transpose()}\)”.
- Section 11.3.5, Proof of Theorem 11.3.12: There is a corrected proof at \url{http://codingthematrix.com/proof-that-first-k-right-singular-vectors-span-closest-space0.pdf}\n
- Section 11.3.10: There is a corrected proof at \url{http://codingthematrix.com/proof-that-U-is-column-orthogonal0.pdf}\n- Task 11.6.6, “To help you debug, applying the procedure to with” should be “To help you debug, applying the procedure with”
- Section 11.4.1: The procedure \texttt{SVD\_solve}(\(A\)) should take the vector \(b\) as a second argument: \texttt{SVD\_solve}(\(A, b\)).
- Section 11.6 (Eigenfaces Lab): \(\{x, y \text{ for } x \text{ in } \text{range(166)} \text{ for } y \text{ in } \text{range(189)}\}\) should be \(\{(x, y) \text{ for } x \text{ in } \text{range(166)} \text{ for } y \text{ in } \text{range(189)}\}\).
- Problem 12.14.8: Error in statement of Lemma 12.14. The eigenvalue of \(A\) having smallest absolute value is the reciprocal of the eigenvalue of \(A^{-1}\) having largest absolute value.
- Section 12.8.1: \(xvec_2^{(t)}\) should be just \(x_2^{(t)}\).
• Section 12.8.1: In the equation
\[
\begin{bmatrix}
x_1^{(t)} \\
x_2^{(t)}
\end{bmatrix} = (S\lambda S^{-1})^t \begin{bmatrix}
x_1^{(0)} \\
x_2^{(0)}
\end{bmatrix}
\]
\(\lambda\) should be \(\Lambda\).

• Section 12.8.1: \(xvec_2(t)\) should be \(x_2^{(t)}\) and \(xvec_2^{(0)}\) should be \(x_2^{(0)}\).

• Section 12.8.4: “Once consecutive addresses have been requested in timesteps \(t\) and \(t+1\), it is very likely that the address requested in timestep \(t+1\) is also consecutive” should end “that the address requested in timestep \(t+2\) is also consecutive.”

• Section 12.12.1: “The theorem in Section 12.8.2...” There is no theorem in that section; the theorem (the Perron-Frobenius Theorem) is not stated in the text.

• Section 12.12.3: The eigenvector given for the test case for Task 12.12.3 is wrong; the correct eigenvector is roughly \(\{1: 0.5222, 2: 0.6182, 3: 0.5738, 4: 0.0705, 5: 0.0783, 6: 0.0705\}\).