Midterm

February 8, 2018

1 Table of Contents

```
In [1]: import pandas as pd
        import numpy as np
        from Bio import SeqIO

        # Graphics
        import matplotlib as mpl
        import matplotlib.pyplot as plt
        import seaborn as sns
        from matplotlib import rc

        rc('text', usetex=True)
        rc('text.latex', preamble=r'\usepackage{cmbright}')
        rc('font', **{'family': 'sans-serif', 'sans-serif': ['Helvetica']})

        # Magic function to make matplotlib inline;
        %matplotlib inline

        # This enables SVG graphics inline.
        # There is a bug, so uncomment if it works.
        %config InlineBackend.figure_formats = {'png', 'retina'}

        # JB's favorite Seaborn settings for notebooks
        rc = {'lines.linewidth': 2,
              'axes.labelsize': 18,
              'axes.titlesize': 18,
              'axes.facecolor': 'DFDFE5'}
        sns.set_context('notebook', rc=rc)
        sns.set_style("dark")

        mpl.rcParams['xtick.labelsize'] = 16
        mpl.rcParams['ytick.labelsize'] = 16
        mpl.rcParams['legend.fontsize'] = 14
```

```
In [2]: def BT(S):
            """Given a string, finds its Burrows-Wheeler transform"""
            def circular_permute(S):
                """Returns an array with all the circular permutations of S"""
                perms = [None]*len(S)
                for i in range(len(S)):
                    pre = S[:i]
                    end = S[i:]
                    row = end + pre
                    perms[i] = row
                return perms

            perms = circular_permute(S)
            perms.sort()

            S_BTed = ''
            for p in perms:
                S_BTed += p[len(S)-1]

            return S_BTed
```

```
In [3]: def inverse_BT(S):
            """Given a Burrows-Wheeler transformed string, finds the original string."""
            for i in range(len(S)):
                if i == 0:
                    cols = sorted(S)
                else:
                    for i in range(len(S)):
                        cols[i] = S[i] + cols[i]
                    cols = sorted(cols)
            for word in cols:
                # this is the line that changed:
                if word[0] == '1':
                    return word
```

```
In [4]: # Check whether this works
        inverse_BT(BT("1BANANANA"))

Out[4]: '1BANANANA'
```

```
In [5]: # invert the provided string:
        inverse_BT('AABBAB1ABBAABBAA')

Out[5]: '1ABABAAABBABBBAA'
```

```
In [ ]:
```