In [ ]:

```
```

Hypothesis Test

In [320]:

```
import numpy as np
import pandas as pd
import scipy.stats as stats
import matplotlib.pyplot as plt
%matplotlib inline
```

In [321]:

```
# Filepath to our excel file.
skookum_data_file = 'CEWA568_data_table.xlsx'

# Use pandas.read_excel() function to open this file.
skookum_data = pd.read_excel(skookum_data_file)

# Now we can see the dataset we loaded:
skookum_data.tail(4)
```

Out[321]:

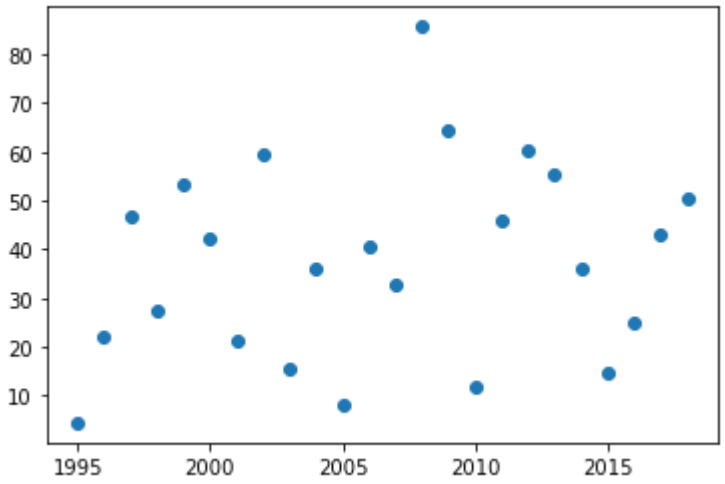
	Year	MaxSWE	Timing of max SWE	SWE value at start of spring melt	Timing of start of spring melt	Timing of end of spring melt	dips	Cumulative Precip (in)	daysofmelt	timing o
20	2015	14.5	2021-12-31	4.0	2021-01-16	2021-01-29	3	130.7	13	13T00:00:00.0
21	2016	24.9	2021-12-30	21.8	2021-03-25	2021-04-24	5	168.9	30	23T00:00:00.0
22	2017	43.0	2021-03-10	41.7	2021-04-12	2021-06-03	5	155.5	52	30T00:00:00.0
23	2018	50.5	2021-04-23	50.5	2021-04-23	2021-06-15	3	172.1	53	21T00:00:00.0

In [322]:

```
# Plot peak streamflows per water year
#fig, ax = plt.subplots(figsize=(7,4))

plt.plot(skookum_data['Year'], skookum_data['MaxSWE'],'o', label= 'ex')
```

Out[322]:

```
[<matplotlib.lines.Line2D at 0x7f1d56cfb910>]
```



We are postulating that there was an change in peak flows around 1975. In other words, how likely is it that the mean of peak flows before 1975 comes from the same distribution as the mean of peak flows after 1975?

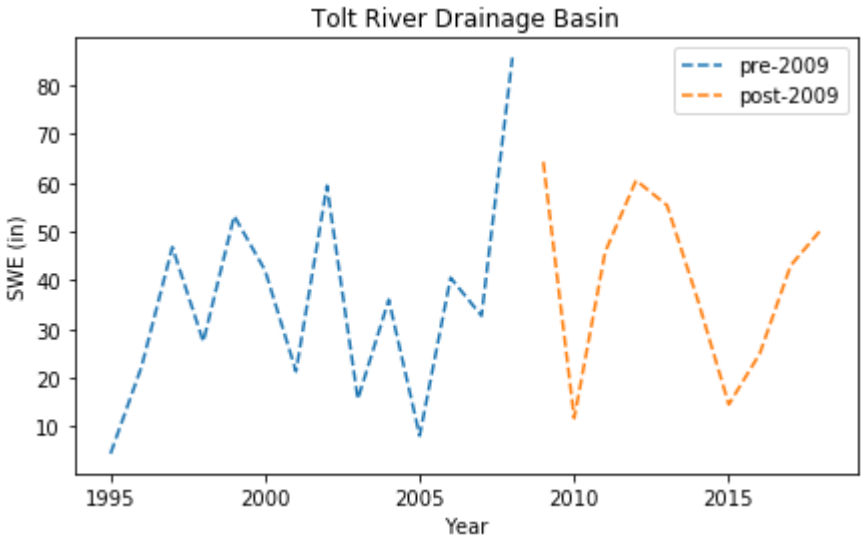
To start, let's split the data in two:

In [323]:

```
# Divide the data into the early period (before 1975) and late period (after and including
1975).

skookum_before = skookum_data[ skookum_data['Year'] < 2009 ]
skookum_after = skookum_data[ skookum_data['Year'] >= 2009 ]
```

In [324]:

```
# Plot our two time periods
fig, ax = plt.subplots(figsize=(7,4))

skookum_before.plot(x='Year', y='MaxSWE', ax=ax, linestyle='--', label='pre-2009')
skookum_after.plot(x='Year', y='MaxSWE', ax=ax, linestyle='--', label='post-2009')

ax.set_ylabel('SWE (in)');
ax.set_title('Tolt River Drainage Basin');
```



What does the distribution of streamflows in each period look like?

Plot a histogram for each period:

fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2, figsize=(10,3))

ax1.hist(skykomish_before['peak value (cfs)'], bins=10) ax1.set_xlim((10000,1.4e5)) ax1.set_xlabel('peak value (cfs)') ax1.set_title('Skykomish River, Annual Peak Streamflow\nHistogram Before 1975')

ax2.hist(skykomish_after['peak value (cfs)'], bins=10) ax2.set_xlim((1e4,1.4e5)) ax2.set_xlabel('peak value (cfs)') ax2.set_title('Skykomish River, Annual Peak Streamflow\nHistogram After 1975');

plt.tight_layout()

Visually compare the distributions of the data, before and after 1975, with theoretical distributions, and random numbers generated from theoretical distributions.

In [325]:

```
### Method 1
# This function requires that the input is a pandas dataframe, with column names, and an i
nteger index
# It returns a copy of the dataframe with an extra column added that has the Cunnane plott
ing positions
def cunnane_quantile(df, column_name):
    '''This function will compute the Cunnane plotting position for the values in a column
of a dataframe.
    It requres a pandas dataframe, and the column name of interest (a text string) as inpu
ts.
    The output is a new dataframe, ranked (sorted) with an extra column with the plotting
 position.
    [Steven Pestana, spestana@uw.edu, Oct. 2020]'''

    # Rank all our values
    ranked_df = df.sort_values(by=[column_name]).reset_index()

    # Calculate the Cunnane plotting position
    ranked_df['cunnane_plotting_position'] = ((ranked_df.index + 1) - (2/5)) / (ranked_df[
column_name].count() + (1/5))

    return ranked_df

### Method 2
# This function should be able to accept any one-dimensional numpy array or list, of numbe
rs
# It returns two numpy arrays, one of the sorted numbers, the other of the plotting positi
on
def cunnane_quantile_array(numbers):
    '''This function also computes the Cunnane plotting position given an array or list of
numbers (rather than a pandas dataframe).
    It has two outputs, first the sorted numbers, second the Cunnane plotting position for
each of those numbers.
    [Steven Pestana, spestana@uw.edu, Oct. 2020]'''

    # 1) sort the data, using the numpy sort function (np.sort())
    sorted_numbers = np.sort(numbers)

    # length of the list of numbers
    n = len(sorted_numbers)

    # make an empty array, of the same length. below we will add the plotting position val
ues to this array
    cunnane_plotting_position = np.empty(n)

    # 2) compute the Cunnane plotting position for each number, using a for loop and the e
numerate function
    for rank, number in enumerate(sorted_numbers):
        cunnane_plotting_position[rank] = ( (rank+1) - (2/5) ) / ( n + (1/5) )

    return sorted_numbers, cunnane_plotting_position
```

In [326]:

```
# Use the cunnane quantile function for before 1975
skookum_before_b = cunnane_quantile(skookum_before, 'MaxSWE')

# Create theoretical normal CDF based on our sample values before 2009
theoretical_cdf_b = stats.norm.cdf(skookum_before_b['MaxSWE'].values,
                                   skookum_before_b['MaxSWE'].mean(),
                                   skookum_before_b['MaxSWE'].std(ddof=1))

# Generate random numbers from a theoretical normal CDF based on our samples before 2009
random_normal_b = np.random.normal(skookum_before_b['MaxSWE'].mean(),
                                   skookum_before_b['MaxSWE'].std(ddof=1),
                                   size=skookum_before_b['MaxSWE'].count())

# Compute the Cunnane plotting position for the random numbers
random_sorted_b, random_quantiles_b  = cunnane_quantile_array(random_normal_b)
```

In [327]:

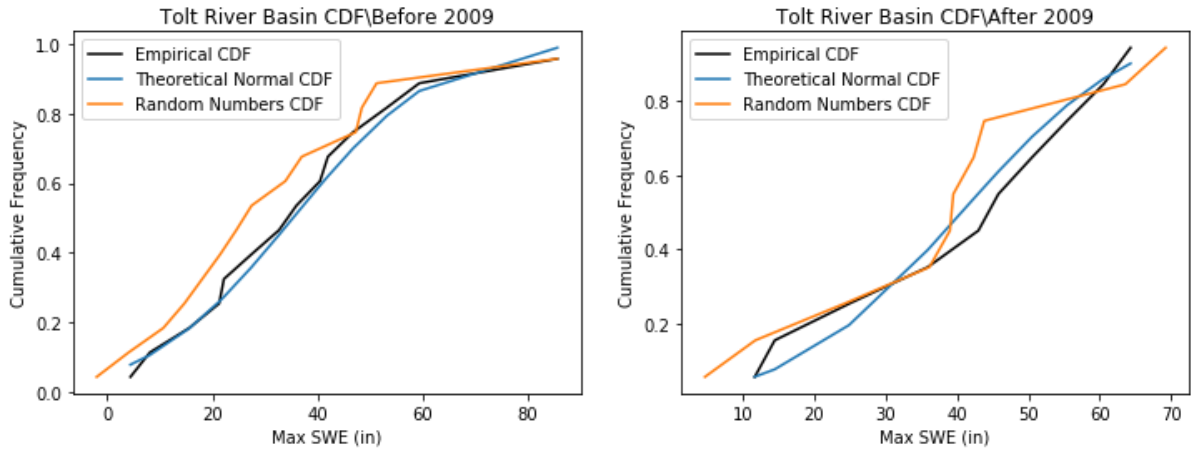
```
# Use the cunnane quantile function for after 1975
skookum_after_a = cunnane_quantile(skookum_after, 'MaxSWE')

# Create theoretical normal CDF based on our sample values before 2009
theoretical_cdf_a = stats.norm.cdf(skookum_after_a['MaxSWE'].values,
                                   skookum_after_a['MaxSWE'].mean(),
                                   skookum_after_a['MaxSWE'].std(ddof=1))

# Generate random numbers from a theoretical normal CDF based on our samples before 2009
random_normal_a = np.random.normal(skookum_after_a['MaxSWE'].mean(),
                                   skookum_after_a['MaxSWE'].std(ddof=1),
                                   size=skookum_after_a['MaxSWE'].count())

# Compute the Cunnane plotting position for the random numbers
random_sorted_a, random_quantiles_a  = cunnane_quantile_array(random_normal_a)
```

In [328]:

```
fig, [ax1, ax2] = plt.subplots(nrows=1, ncols=2, figsize=(12,4))

# Before 2009
# Empirical CDF
ax1.plot(skookum_before_b['MaxSWE'], skookum_before_b['cunnane_plotting_position'], color=
'k', label='Empirical CDF')
# Theorectical Normal CDF
ax1.plot(skookum_before_b['MaxSWE'], theoretical_cdf_b, label='Theoretical Normal CDF')
# Random numbers CDF from a theoretical normal distribution
ax1.plot(random_sorted_b,random_quantiles_b,'-', label='Random Numbers CDF')
# Add legend and labels
ax1.legend()
ax1.set_ylabel('Cumulative Frequency')
ax1.set_xlabel('Max SWE (in)')
ax1.set_title('Tolt River Basin CDF\Before 2009')

# After 2009
# Empirical CDF
ax2.plot(skookum_after_a['MaxSWE'], skookum_after_a['cunnane_plotting_position'], color=
'k', label='Empirical CDF')
# Theorectical Normal CDF
ax2.plot(skookum_after_a['MaxSWE'], theoretical_cdf_a, label='Theoretical Normal CDF')
# Random numbers CDF from a theoretical normal distribution
ax2.plot(random_sorted_a, random_quantiles_a,'-', label='Random Numbers CDF')
# Add legend and labels
ax2.legend()
ax2.set_ylabel('Cumulative Frequency')
ax2.set_xlabel('Max SWE (in)')
ax2.set_title('Tolt River Basin CDF\After 2009')
```

Out[328]:

```
Text(0.5, 1.0, 'Tolt River Basin CDF\\After 2009')
```



Does the streamflow data look normally distributed? Maybe try changing the above code to compare the empirical CDFs against theoretical lognormal distributions. (Remember to transform the mean and standard deviations into "log space")

Two-Sample Z-Test

Returning to our question: We are postulating (making a hypothesis) that there was a change in the mean flood statistics after 1975, and we want to test whether this is true. Where do we start?

First we can formally state our null hypothesis, and our alternative hypothesis. We are also told to use a two sample test, and to set α at 5%.

Our **null hypothesis** is that the peak flows of the early period ($\bar{X}_1$) are drawn from the same distribution as the peak flows of the later period ($\bar{X}_2$) (therefore the distributions means of the two time periods are equal):

$H_0: \bar{X}_1 = \bar{X}_2$

Our **alternative hypothesis** is that the mean of the distribution for the later period is greater than that of the early period:

$H_1: \bar{X}_2 > \bar{X}_1$

We can also state these as:

$H_0: \bar{X}_1 - \bar{X}_2 = \mu_0$

$H_1: \bar{X}_1 - \bar{X}_2 < \mu_0$

Where μ_0 is the hypothesized difference between the population means, and in this case $\mu_0 = \mu_1

Note that I have written a "one-sided (https://en.wikipedia.org/wiki/One-_and_two-tailed_tests)" test here because we are testing only for a change in one direction (an increase). We think that either the mean flood increased or it didn't change; we do not think the mean flood decreased:

- This might be chosen because we have some physical reason to think it increased (e.g. higher elevations in the watershed now get rainfall where it used to mostly get snow because of our warming climate).
- Or this might be chosen because we have some practical reason for the test to matter in this particular direction (e.g. we will change flood zoning downstream and/or how we operate a reservoir if the mean flood has increased, but won't make a change if it decreased).

But which test should we use? Is the z-distribution valid?

We are using the z-test (https://en.wikipedia.org/wiki/Z-test), which uses the standard normal distribution. From our work above, we know that our data are likely not neccesarily normally distributed. However, the central limit theorem (https://en.wikipedia.org/wiki/Central_limit_theorem) tells us that, "*If a sample of n values is extracted at random from a population with mean μ and standard deviation σ, and n > 30, then the means of these samples are approximately normally distributed*"

We calculate our z-score as: $\displaystyle Z = \frac{ (\bar{X}_2 - \bar{X}_1) - \mu _{0} } { s_{1,2} }$

Where $s_{1,2}$ is the "pooled standard deviation", s_1, s_2 and n_1, n_2 are the two standard deviations and sample sizes respectively.

$s_{1,2} = \displaystyle\sqrt{ \displaystyle\frac{s^2_1}{n_1} + \displaystyle\frac{s^2_2}{n_2} }$

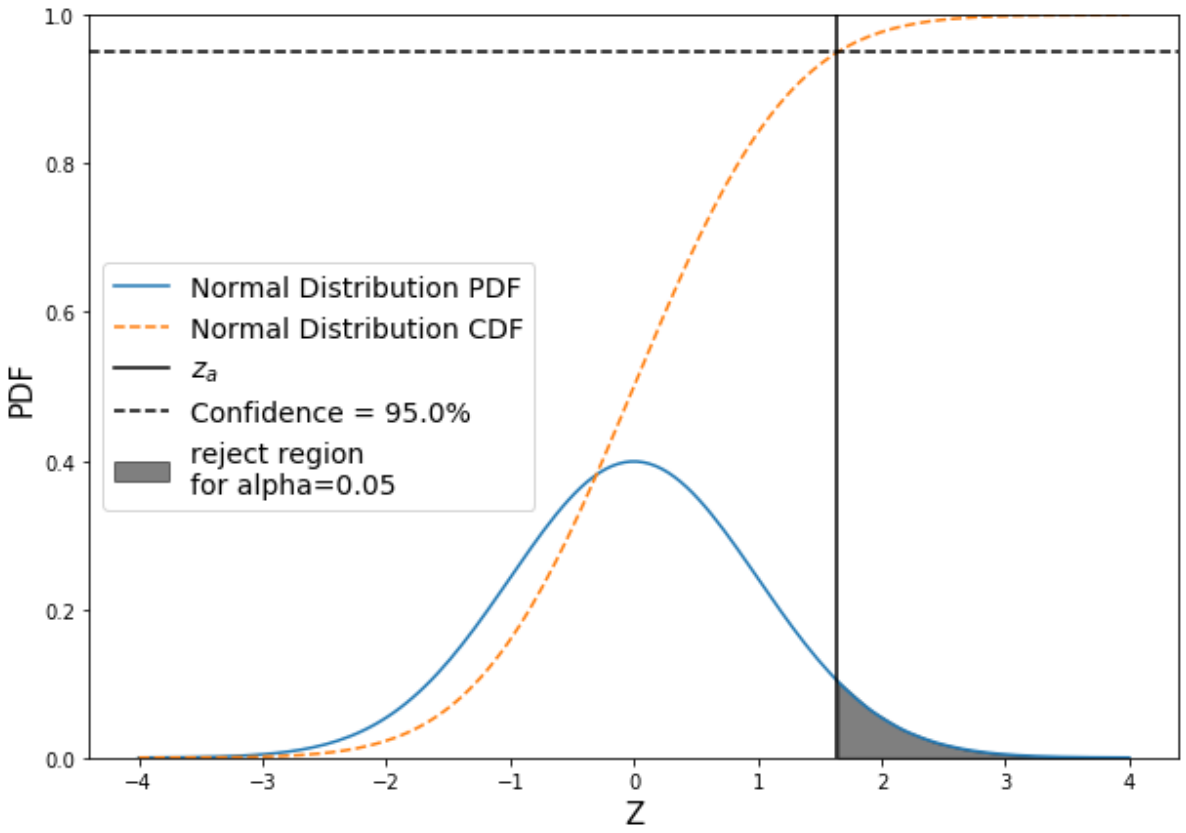
Remember, the means are normally distributed even if the data themselves are not normally distributed.

So what does the "**Null Distribution** look like?

And what do the "rejection regions" look like?

In [329]:

```
fig, ax = plt.subplots(figsize=(10,7))

# Create a null pdf
x = np.linspace(-4, 4, num=160)
ax.plot(x, stats.norm.pdf(x, 0, 1), label='Normal Distribution PDF')

# Plot the null cdf
ax.plot(x, stats.norm.cdf(x, 0, 1), linestyle='--', label='Normal Distribution CDF')

# Plot the region that z_test would have to fall in in order for us to reject the null hyp
othesis
conf = 0.95
z_alpha = stats.norm.ppf(conf)
shade = np.linspace(z_alpha, 4, 10)
ax.fill_between(shade, stats.norm.pdf(shade, 0, 1) ,  color='k', alpha=0.5, label='reject
 region\nfor alpha={}'.format(np.round(1-conf,2)))
# Plot a line at z_alpha
plt.axvline(z_alpha, color='black', label='$z_{a}$')
# Plot a line at our 95% confidence
plt.axhline(conf, color='black', linestyle='--', label='Confidence = {}%'.format(conf*100
))


# Add labels
ax.set_ylim((0,1))
plt.xlabel('Z', fontsize=15)
plt.ylabel('PDF', fontsize=15)
ax.legend(loc='center left', fontsize=14);
```



In [330]:

```
# Check that we have a large enough sample size (n>30)

n = len(skookum_before['MaxSWE'])
print(n)

m = len(skookum_after['MaxSWE'])
print(m)
```

```
14
10
```

Both are larger than 30, so we can go ahead and calculate the z-score for our test:

In [331]:

```
# We want out alpha to be 0.05
alpha = 0.05
# This gives us a confidence of 0.95
conf = 1 - alpha
```

Next, determine which value in the z-distribution corresponds to our 0.95 confidence in the CDF

In [332]:

```
z_alpha = stats.norm.ppf(conf)
print("z_alpha = {}".format(z_alpha))
```

```
z_alpha = 1.6448536269514722
```

Compute the pooled standard deviation, $s_{1,2} = \displaystyle\sqrt{ \displaystyle\frac{s^2_1}{n_1} + \displaystyle\frac{s^2_2}{n_2} }$

In [333]:

```
# Compute the pooled standard deviaiton
pooled_sd = np.sqrt( skookum_before['MaxSWE'].std(ddof=1)**2 / n + skookum_after['MaxSWE']
.std(ddof=1)**2 / m )
```

Finally, compute our z-score as $\displaystyle Z = \frac{ (\bar{X}_2 - \bar{X}_1) - \mu _{0} } { s_{1,2} }$

In [334]:

```
# hypothesizing no change
mu_0 = 0

# compute z-score
zscore = (skookum_after['MaxSWE'].mean() - skookum_before['MaxSWE'].mean() - mu_0)/pooled_
sd

print("z-score = {}".format( np.round(zscore,2) ))
```

```
z-score = 0.64
```

We can also compute a p-value from this z-score by looking it up on the standard normal distribution CDF

In [335]:

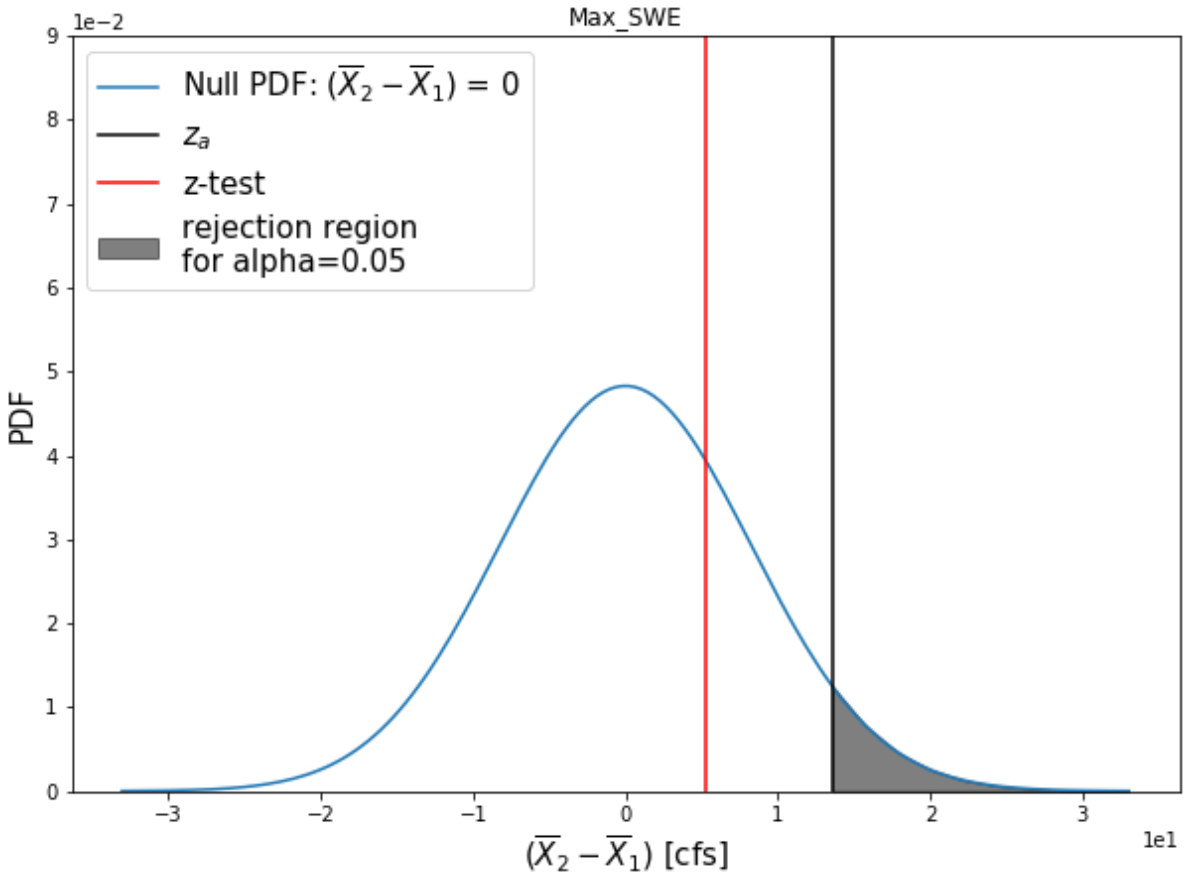
```
pvalue = 1 - stats.norm.cdf(zscore)
print("p = {}".format( np.round(pvalue,3) ))
```

```
p = 0.262
```

Plot the z-distribution, our z-score test result, and the z_α that corresponds with our 95% confidence interval.

In [336]:

```
# Create z values between -4 and 4 to look at the middle portion of the z-distribution aro
und 0
# Scale our values by the pooled standard deviation (otherwise we'd be in generic z-distri
bution space)
z = np.linspace(-4, 4, num=160) * pooled_sd

# Create the plot
plt.figure(figsize=(10,7))
# Plot the z-distribution here
plt.plot(z, stats.norm.pdf(z, 0, pooled_sd), label='Null PDF: ($\overline{X}_2 - \overline
{X}_1$) = 0')

# Plot a line at our z-alpha value and shade the rejection region
plt.axvline(z_alpha*pooled_sd, color='black', linestyle='-', label='$z_{a}$')
shade = np.linspace(z_alpha*pooled_sd, np.max(z), 10)
plt.fill_between(shade, stats.norm.pdf(shade, 0, pooled_sd) ,  color='k', alpha=0.5, label
='rejection region\nfor alpha={}'.format(np.round(1-conf,2)))


plt.axvline(zscore*pooled_sd, color='red', linestyle='-', label='z-test')
plt.xlabel('($\overline{X}_2 - \overline{X}_1$) [cfs]', fontsize=15)
plt.ylabel('PDF', fontsize=15)
plt.title('Max_SWE')
plt.ticklabel_format(axis='x', style='sci', scilimits=(0,0))
plt.ticklabel_format(axis='y', style='sci', scilimits=(0,0))
plt.ylim(0, 9e-2)
plt.legend(loc='best', fontsize=15);
```



Repeat for other variables

DIPS

In [337]:

```
# Use the cunnane quantile function for before 1975
skookum_before_b = cunnane_quantile(skookum_before, 'dips')

# Create theoretical normal CDF based on our sample values before 2009
theoretical_cdf_b = stats.norm.cdf(skookum_before_b['dips'].values,
                                   skookum_before_b['dips'].mean(),
                                   skookum_before_b['dips'].std(ddof=1))

# Generate random numbers from a theoretical normal CDF based on our samples before 2009
random_normal_b = np.random.normal(skookum_before_b['dips'].mean(),
                                   skookum_before_b['dips'].std(ddof=1),
                                   size=skookum_before_b['dips'].count())

# Compute the Cunnane plotting position for the random numbers
random_sorted_b, random_quantiles_b  = cunnane_quantile_array(random_normal_b)
```

In [338]:

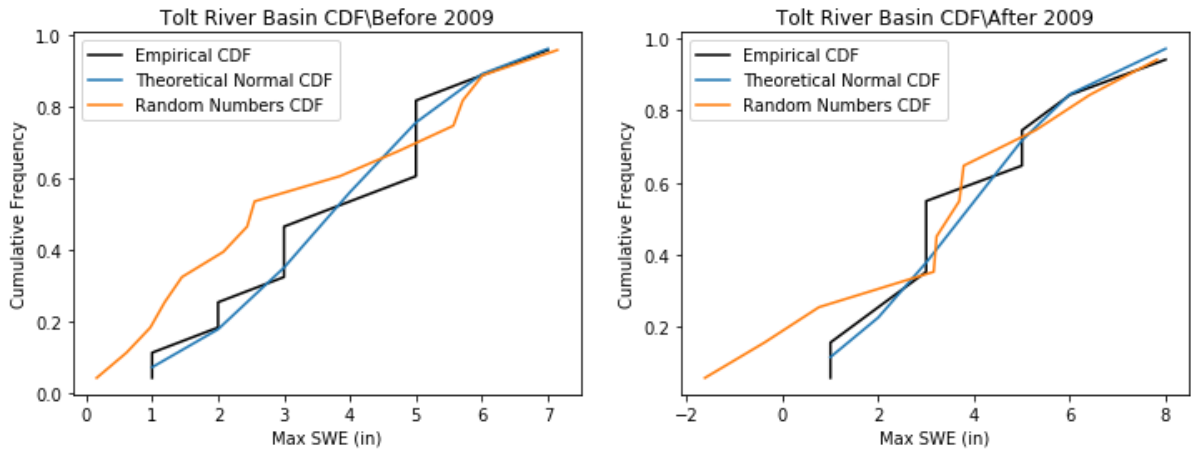
```
# Use the cunnane quantile function for after 1975
skookum_after_a = cunnane_quantile(skookum_after, 'dips')

# Create theoretical normal CDF based on our sample values before 2009
theoretical_cdf_a = stats.norm.cdf(skookum_after_a['dips'].values,
                                   skookum_after_a['dips'].mean(),
                                   skookum_after_a['dips'].std(ddof=1))

# Generate random numbers from a theoretical normal CDF based on our samples before 2009
random_normal_a = np.random.normal(skookum_after_a['dips'].mean(),
                                   skookum_after_a['dips'].std(ddof=1),
                                   size=skookum_after_a['dips'].count())

# Compute the Cunnane plotting position for the random numbers
random_sorted_a, random_quantiles_a  = cunnane_quantile_array(random_normal_a)
```

In [339]:

```
fig, [ax1, ax2] = plt.subplots(nrows=1, ncols=2, figsize=(12,4))

# Before 2009
# Empirical CDF
ax1.plot(skookum_before_b['dips'], skookum_before_b['cunnane_plotting_position'], color=
'k', label='Empirical CDF')
# Theorectical Normal CDF
ax1.plot(skookum_before_b['dips'], theoretical_cdf_b, label='Theoretical Normal CDF')
# Random numbers CDF from a theoretical normal distribution
ax1.plot(random_sorted_b,random_quantiles_b,'-', label='Random Numbers CDF')
# Add legend and labels
ax1.legend()
ax1.set_ylabel('Cumulative Frequency')
ax1.set_xlabel('Max SWE (in)')
ax1.set_title('Tolt River Basin CDF\Before 2009')

# After 2009
# Empirical CDF
ax2.plot(skookum_after_a['dips'], skookum_after_a['cunnane_plotting_position'], color='k',
label='Empirical CDF')
# Theorectical Normal CDF
ax2.plot(skookum_after_a['dips'], theoretical_cdf_a, label='Theoretical Normal CDF')
# Random numbers CDF from a theoretical normal distribution
ax2.plot(random_sorted_a, random_quantiles_a,'-', label='Random Numbers CDF')
# Add legend and labels
ax2.legend()
ax2.set_ylabel('Cumulative Frequency')
ax2.set_xlabel('Max SWE (in)')
ax2.set_title('Tolt River Basin CDF\After 2009')
```

Out[339]:

```
Text(0.5, 1.0, 'Tolt River Basin CDF\\After 2009')
```



Does the streamflow data look normally distributed? Maybe try changing the above code to compare the empirical CDFs against theoretical lognormal distributions. (Remember to transform the mean and standard deviations into "log space")

Two-Sample Z-Test

Returning to our question: We are postulating (making a hypothesis) that there was a change in the mean flood statistics after 1975, and we want to test whether this is true. Where do we start?

First we can formally state our null hypothesis, and our alternative hypothesis. We are also told to use a two sample test, and to set α at 5%.

Our **null hypothesis** is that the peak flows of the early period ($\bar{X}_1$) are drawn from the same distribution as the peak flows of the later period ($\bar{X}_2$) (therefore the distributions means of the two time periods are equal):

$H_0: \bar{X}_1 = \bar{X}_2$

Our **alternative hypothesis** is that the mean of the distribution for the later period is greater than that of the early period:

$H_1: \bar{X}_2 > \bar{X}_1$

We can also state these as:

$H_0: \bar{X}_1 - \bar{X}_2 = \mu_0$

$H_1: \bar{X}_1 - \bar{X}_2 < \mu_0$

Where μ_0 is the hypothesized difference between the population means, and in this case $\mu_0 = \mu_1 - \mu_2 = 0$

Note that I have written a "one-sided (https://en.wikipedia.org/wiki/One-_and_two-tailed_tests)" test here because we are testing only for a change in one direction (an increase). We think that either the mean flood increased or it didn't change; we do not think the mean flood decreased:

- This might be chosen because we have some physical reason to think it increased (e.g. higher elevations in the watershed now get rainfall where it used to mostly get snow because of our warming climate).
- Or this might be chosen because we have some practical reason for the test to matter in this particular direction (e.g. we will change flood zoning downstream and/or how we operate a reservoir if the mean flood has increased, but won't make a change if it decreased).

But which test should we use? Is the z-distribution valid?

We are using the z-test (https://en.wikipedia.org/wiki/Z-test), which uses the standard normal distribution. From our work above, we know that our data are likely not neccesarily normally distributed. However, the central limit theorem (https://en.wikipedia.org/wiki/Central_limit_theorem) tells us that, "*If a sample of n values is extracted at random from a population with mean μ and standard deviation σ, and n > 30, then the means of these samples are approximately normally distributed*"

We calculate our z-score as: $\displaystyle Z = \frac{ (\bar{X}_2 - \bar{X}_1) - \mu _{0} } { s_{1,2} }$

Where $s_{1,2}$ is the "pooled standard deviation", s_1, s_2 and n_1, n_2 are the two standard deviations and sample sizes respectively.

$s_{1,2} = \displaystyle\sqrt{ \displaystyle\frac{s^2_1}{n_1} + \displaystyle\frac{s^2_2}{n_2} }$

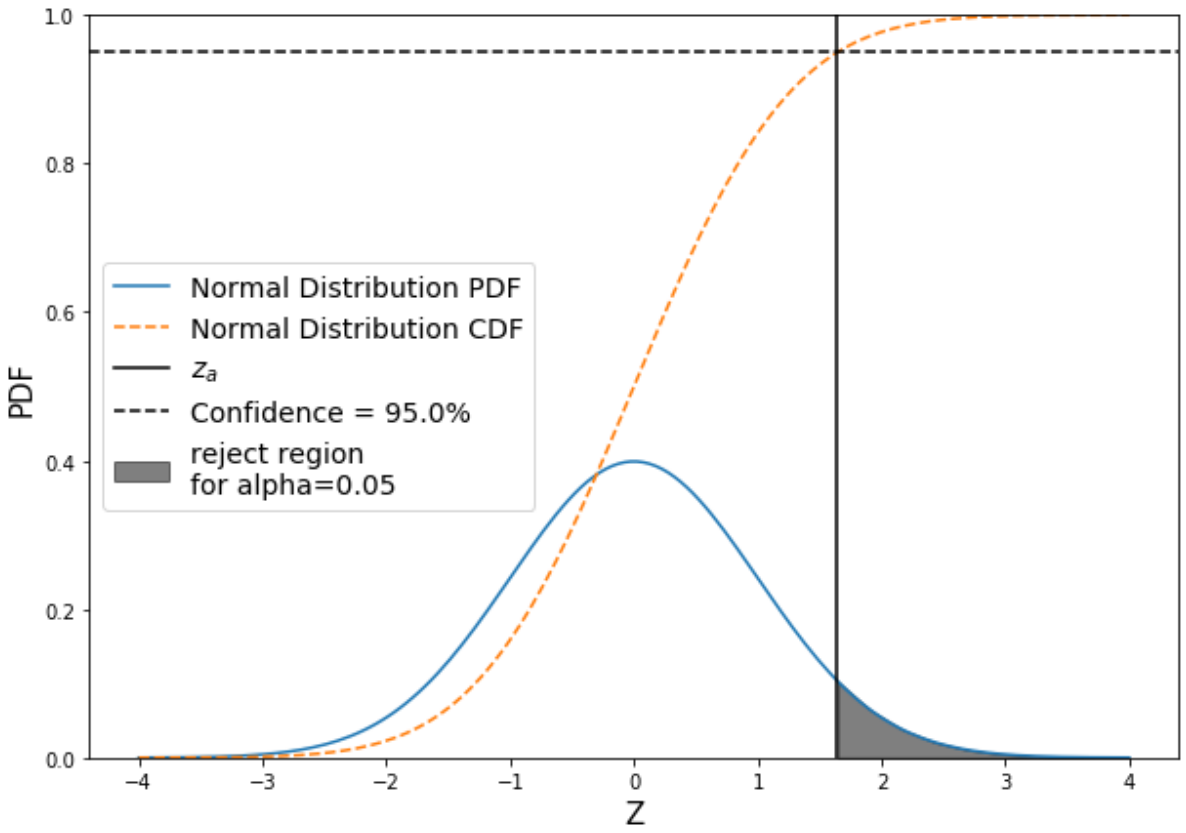
Remember, the means are normally distributed even if the data themselves are not normally distributed.

So what does the "**Null Distribution** look like?

And what do the "rejection regions" look like?

In [340]:

```
fig, ax = plt.subplots(figsize=(10,7))

# Create a null pdf
x = np.linspace(-4, 4, num=160)
ax.plot(x, stats.norm.pdf(x, 0, 1), label='Normal Distribution PDF')

# Plot the null cdf
ax.plot(x, stats.norm.cdf(x, 0, 1), linestyle='--', label='Normal Distribution CDF')

# Plot the region that z_test would have to fall in in order for us to reject the null hyp
othesis
conf = 0.95
z_alpha = stats.norm.ppf(conf)
shade = np.linspace(z_alpha, 4, 10)
ax.fill_between(shade, stats.norm.pdf(shade, 0, 1) ,  color='k', alpha=0.5, label='reject
 region\nfor alpha={}'.format(np.round(1-conf,2)))
# Plot a line at z_alpha
plt.axvline(z_alpha, color='black', label='$z_{a}$')
# Plot a line at our 95% confidence
plt.axhline(conf, color='black', linestyle='--', label='Confidence = {}%'.format(conf*100
))


# Add labels
ax.set_ylim((0,1))
plt.xlabel('Z', fontsize=15)
plt.ylabel('PDF', fontsize=15)
ax.legend(loc='center left', fontsize=14);
```



In [341]:

```
# Check that we have a large enough sample size (n>30)

n = len(skookum_before['dips'])
print(n)

m = len(skookum_after['dips'])
print(m)
```

```
14
10
```

Both are larger than 30, so we can go ahead and calculate the z-score for our test:

In [342]:

```
# We want out alpha to be 0.05
alpha = 0.05
# This gives us a confidence of 0.95
conf = 1 - alpha
```

Next, determine which value in the z-distribution corresponds to our 0.95 confidence in the CDF

In [343]:

```
z_alpha = stats.norm.ppf(conf)
print("z_alpha = {}".format(z_alpha))
```

```
z_alpha = 1.6448536269514722
```

Compute the pooled standard deviation, $s_{1,2} = \displaystyle\sqrt{ \displaystyle\frac{s^2_1}{n_1} + \displaystyle\frac{s^2_2}{n_2} }$

In [344]:

```
# Compute the pooled standard deviaiton
pooled_sd = np.sqrt( skookum_before['dips'].std(ddof=1)**2 / n + skookum_after['dips'].std
(ddof=1)**2 / m )
```

Finally, compute our z-score as $\displaystyle Z = \frac{ (\bar{X}_2 - \bar{X}_1) - \mu _{0} } { s_{1,2} }$

In [345]:

```
# hypothesizing no change
mu_0 = 0

# compute z-score
zscore = (skookum_after['dips'].mean() - skookum_before['dips'].mean() - mu_0)/pooled_sd

print("z-score = {}".format( np.round(zscore,2) ))
```

```
z-score = -0.02
```

We can also compute a p-value from this z-score by looking it up on the standard normal distribution CDF

In [346]:

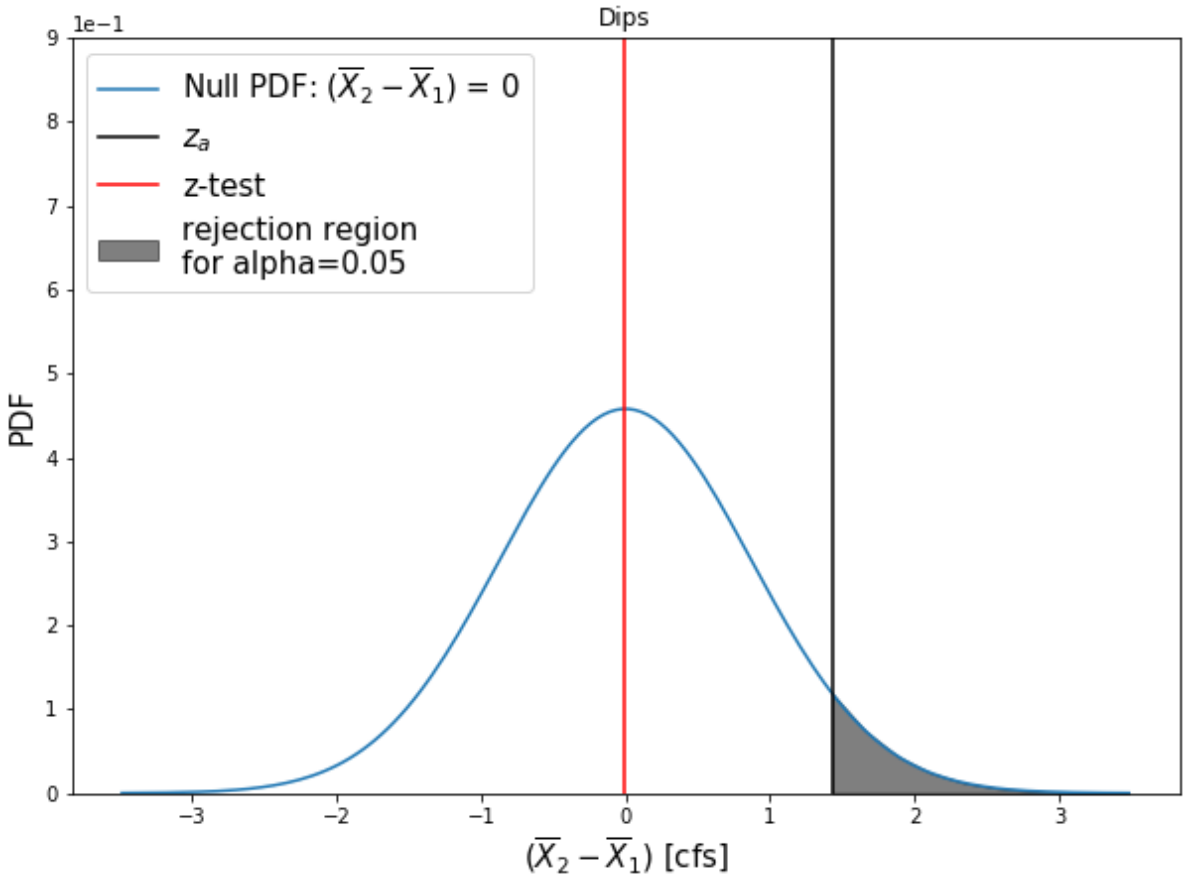
```
pvalue = 1 - stats.norm.cdf(zscore)
print("p = {}".format( np.round(pvalue,3) ))
```

```
p = 0.507
```

Plot the z-distribution, our z-score test result, and the z_α that corresponds with our 95% confidence interval.

In [347]:

```
# Create z values between -4 and 4 to look at the middle portion of the z-distribution aro
und 0
# Scale our values by the pooled standard deviation (otherwise we'd be in generic z-distri
bution space)
z = np.linspace(-4, 4, num=160) * pooled_sd

# Create the plot
plt.figure(figsize=(10,7))
# Plot the z-distribution here
plt.plot(z, stats.norm.pdf(z, 0, pooled_sd), label='Null PDF: ($\overline{X}_2 - \overline
{X}_1$) = 0')

# Plot a line at our z-alpha value and shade the rejection region
plt.axvline(z_alpha*pooled_sd, color='black', linestyle='-', label='$z_{a}$')
shade = np.linspace(z_alpha*pooled_sd, np.max(z), 10)
plt.fill_between(shade, stats.norm.pdf(shade, 0, pooled_sd) ,  color='k', alpha=0.5, label
='rejection region\nfor alpha={}'.format(np.round(1-conf,2)))


plt.axvline(zscore*pooled_sd, color='red', linestyle='-', label='z-test')
plt.xlabel('($\overline{X}_2 - \overline{X}_1$) [cfs]', fontsize=15)
plt.ylabel('PDF', fontsize=15)
plt.title('Dips')
plt.ticklabel_format(axis='x', style='sci', scilimits=(0,0))
plt.ticklabel_format(axis='y', style='sci', scilimits=(0,0))
plt.ylim(0, 9e-1)
plt.legend(loc='best', fontsize=15);
```


Dips
1e−1
Null PDF: $(\overline{X}_2 - \overline{X}_1) = 0$
z_a
z-test
rejection region
for alpha=0.05
PDF
$(\overline{X}_2 - \overline{X}_1)$ [cfs]

Days of Melt

In [348]:

```
# Use the cunnane quantile function for before 1975
skookum_before_b = cunnane_quantile(skookum_before, 'daysofmelt')

# Create theoretical normal CDF based on our sample values before 2009
theoretical_cdf_b = stats.norm.cdf(skookum_before_b['daysofmelt'].values,
                                   skookum_before_b['daysofmelt'].mean(),
                                   skookum_before_b['daysofmelt'].std(ddof=1))

# Generate random numbers from a theoretical normal CDF based on our samples before 2009
random_normal_b = np.random.normal(skookum_before_b['daysofmelt'].mean(),
                                   skookum_before_b['daysofmelt'].std(ddof=1),
                                   size=skookum_before_b['daysofmelt'].count())

# Compute the Cunnane plotting position for the random numbers
random_sorted_b, random_quantiles_b  = cunnane_quantile_array(random_normal_b)
```

In [349]:

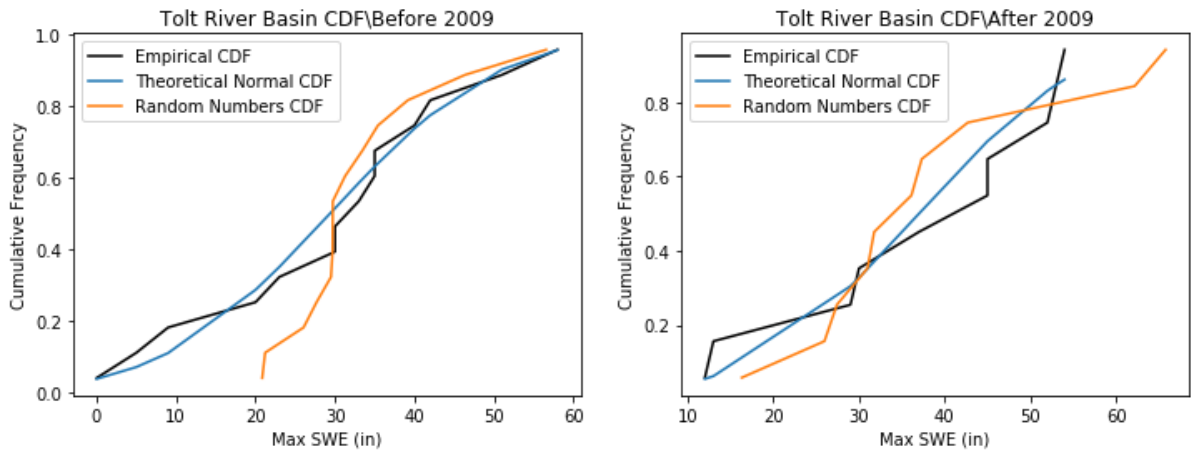
```
# Use the cunnane quantile function for after 1975
skookum_after_a = cunnane_quantile(skookum_after, 'daysofmelt')

# Create theoretical normal CDF based on our sample values before 2009
theoretical_cdf_a = stats.norm.cdf(skookum_after_a['daysofmelt'].values,
                                   skookum_after_a['daysofmelt'].mean(),
                                   skookum_after_a['daysofmelt'].std(ddof=1))

# Generate random numbers from a theoretical normal CDF based on our samples before 2009
random_normal_a = np.random.normal(skookum_after_a['daysofmelt'].mean(),
                                   skookum_after_a['daysofmelt'].std(ddof=1),
                                   size=skookum_after_a['daysofmelt'].count())

# Compute the Cunnane plotting position for the random numbers
random_sorted_a, random_quantiles_a  = cunnane_quantile_array(random_normal_a)
```

In [350]:

```
fig, [ax1, ax2] = plt.subplots(nrows=1, ncols=2, figsize=(12,4))

# Before 2009
# Empirical CDF
ax1.plot(skookum_before_b['daysofmelt'], skookum_before_b['cunnane_plotting_position'], co
lor='k', label='Empirical CDF')
# Theorectical Normal CDF
ax1.plot(skookum_before_b['daysofmelt'], theoretical_cdf_b, label='Theoretical Normal CDF'
)
# Random numbers CDF from a theoretical normal distribution
ax1.plot(random_sorted_b,random_quantiles_b,'-', label='Random Numbers CDF')
# Add legend and labels
ax1.legend()
ax1.set_ylabel('Cumulative Frequency')
ax1.set_xlabel('Max SWE (in)')
ax1.set_title('Tolt River Basin CDF\Before 2009')

# After 2009
# Empirical CDF
ax2.plot(skookum_after_a['daysofmelt'], skookum_after_a['cunnane_plotting_position'], colo
r='k', label='Empirical CDF')
# Theorectical Normal CDF
ax2.plot(skookum_after_a['daysofmelt'], theoretical_cdf_a, label='Theoretical Normal CDF')
# Random numbers CDF from a theoretical normal distribution
ax2.plot(random_sorted_a, random_quantiles_a,'-', label='Random Numbers CDF')
# Add legend and labels
ax2.legend()
ax2.set_ylabel('Cumulative Frequency')
ax2.set_xlabel('Max SWE (in)')
ax2.set_title('Tolt River Basin CDF\After 2009')
```

Out[350]:

```
Text(0.5, 1.0, 'Tolt River Basin CDF\\After 2009')
```



Does the streamflow data look normally distributed? Maybe try changing the above code to compare the empirical CDFs against theoretical lognormal distributions. (Remember to transform the mean and standard deviations into "log space")

Two-Sample Z-Test

Returning to our question: We are postulating (making a hypothesis) that there was a change in the mean flood statistics after 1975, and we want to test whether this is true. Where do we start?

First we can formally state our null hypothesis, and our alternative hypothesis. We are also told to use a two sample test, and to set α at 5%.

Our **null hypothesis** is that the peak flows of the early period ($\bar{X}_1$) are drawn from the same distribution as the peak flows of the later period ($\bar{X}_2$) (therefore the distributions means of the two time periods are equal):

$H_0: \bar{X}_1 = \bar{X}_2$

Our **alternative hypothesis** is that the mean of the distribution for the later period is greater than that of the early period:

$H_1: \bar{X}_2 > \bar{X}_1$

We can also state these as:

$H_0: \bar{X}_1 - \bar{X}_2 = \mu_0$

$H_1: \bar{X}_1 - \bar{X}_2 < \mu_0$

Where μ_0 is the hypothesized difference between the population means, and in this case $\mu_0 = \mu_1 - \mu_2 = 0$

Note that I have written a "one-sided (https://en.wikipedia.org/wiki/One-_and_two-tailed_tests)" test here because we are testing only for a change in one direction (an increase). We think that either the mean flood increased or it didn't change; we do not think the mean flood decreased:

- This might be chosen because we have some physical reason to think it increased (e.g. higher elevations in the watershed now get rainfall where it used to mostly get snow because of our warming climate).
- Or this might be chosen because we have some practical reason for the test to matter in this particular direction (e.g. we will change flood zoning downstream and/or how we operate a reservoir if the mean flood has increased, but won't make a change if it decreased).

But which test should we use? Is the z-distribution valid?

We are using the z-test (https://en.wikipedia.org/wiki/Z-test), which uses the standard normal distribution. From our work above, we know that our data are likely not neccesarily normally distributed. However, the central limit theorem (https://en.wikipedia.org/wiki/Central_limit_theorem) tells us that, "*If a sample of n values is extracted at random from a population with mean μ and standard deviation σ, and n > 30, then the means of these samples are approximately normally distributed*"

We calculate our z-score as: $\displaystyle Z = \frac{ (\bar{X}_2 - \bar{X}_1) - \mu _{0} } { s_{1,2} }$

Where $s_{1,2}$ is the "pooled standard deviation", s_1, s_2 and n_1, n_2 are the two standard deviations and sample sizes respectively.

$s_{1,2} = \displaystyle\sqrt{ \displaystyle\frac{s^2_1}{n_1} + \displaystyle\frac{s^2_2}{n_2} }$

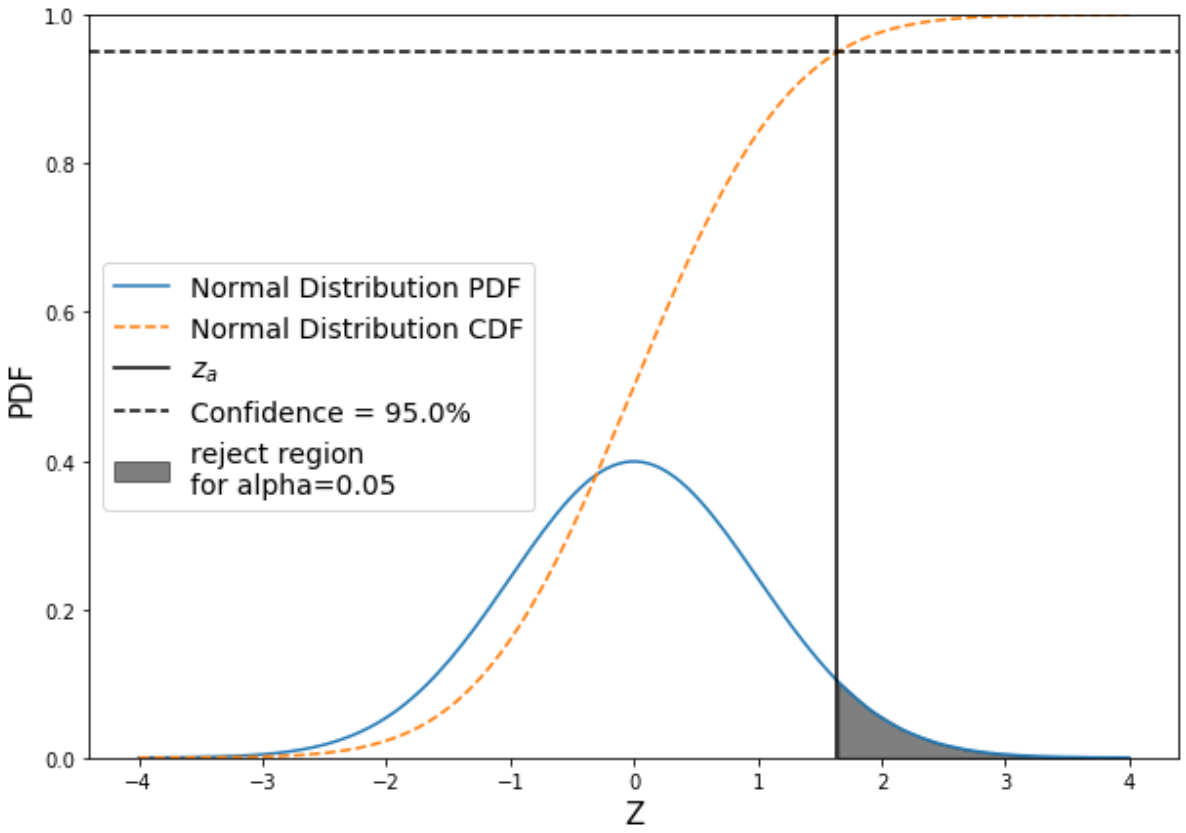
Remember, the means are normally distributed even if the data themselves are not normally distributed.

So what does the "**Null Distribution** look like?

And what do the "rejection regions" look like?

In [351]:

```
fig, ax = plt.subplots(figsize=(10,7))

# Create a null pdf
x = np.linspace(-4, 4, num=160)
ax.plot(x, stats.norm.pdf(x, 0, 1), label='Normal Distribution PDF')

# Plot the null cdf
ax.plot(x, stats.norm.cdf(x, 0, 1), linestyle='--', label='Normal Distribution CDF')

# Plot the region that z_test would have to fall in in order for us to reject the null hyp
othesis
conf = 0.95
z_alpha = stats.norm.ppf(conf)
shade = np.linspace(z_alpha, 4, 10)
ax.fill_between(shade, stats.norm.pdf(shade, 0, 1) ,  color='k', alpha=0.5, label='reject
 region\nfor alpha={}'.format(np.round(1-conf,2)))
# Plot a line at z_alpha
plt.axvline(z_alpha, color='black', label='$z_{a}$')
# Plot a line at our 95% confidence
plt.axhline(conf, color='black', linestyle='--', label='Confidence = {}%'.format(conf*100
))


# Add labels
ax.set_ylim((0,1))
plt.xlabel('Z', fontsize=15)
plt.ylabel('PDF', fontsize=15)
ax.legend(loc='center left', fontsize=14);
```



In [352]:

```
# Check that we have a large enough sample size (n>30)

n = len(skookum_before['daysofmelt'])
print(n)

m = len(skookum_after['daysofmelt'])
print(m)
```

```
14
10
```

Both are larger than 30, so we can go ahead and calculate the z-score for our test:

In [353]:

```
# We want out alpha to be 0.05
alpha = 0.05
# This gives us a confidence of 0.95
conf = 1 - alpha
```

Next, determine which value in the z-distribution corresponds to our 0.95 confidence in the CDF

In [354]:

```
z_alpha = stats.norm.ppf(conf)
print("z_alpha = {}".format(z_alpha))
```

```
z_alpha = 1.6448536269514722
```

Compute the pooled standard deviation, $s_{1,2} = \displaystyle\sqrt{ \displaystyle\frac{s^2_1}{n_1} + \displaystyle\frac{s^2_2}{n_2} }$

In [355]:

```
# Compute the pooled standard deviaiton
pooled_sd = np.sqrt( skookum_before['daysofmelt'].std(ddof=1)**2 / n + skookum_after['daysofmelt'].std(ddof=1)**2 / m )
```

Finally, compute our z-score as $\displaystyle Z = \frac{ (\bar{X}_2 - \bar{X}_1) - \mu _{0} } { s_{1,2} }$

In [356]:

```
# hypothesizing no change
mu_0 = 0

# compute z-score
zscore = (skookum_after['daysofmelt'].mean() - skookum_before['daysofmelt'].mean() - mu_0)
/pooled_sd

print("z-score = {}".format( np.round(zscore,2) ))
```

```
z-score = 1.14
```

We can also compute a p-value from this z-score by looking it up on the standard normal distribution CDF

In [357]:

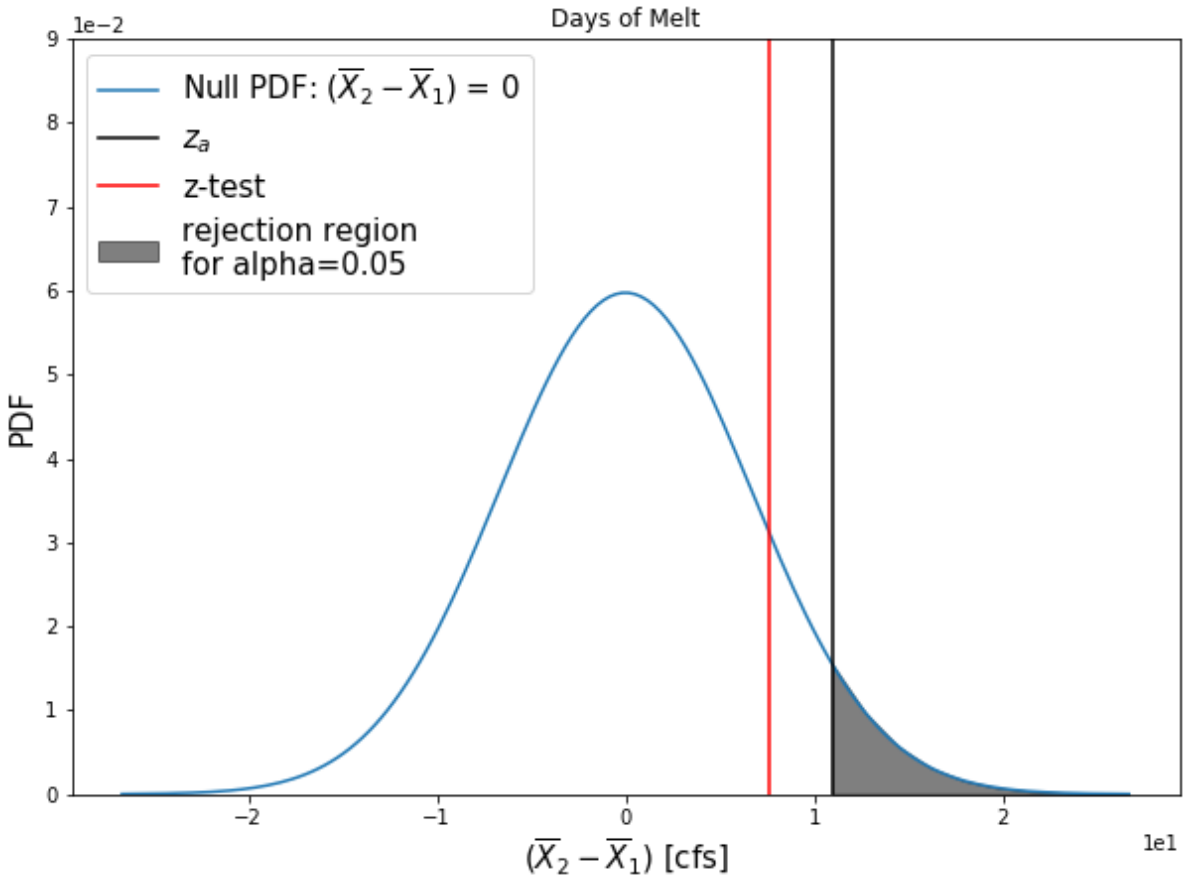
```
pvalue = 1 - stats.norm.cdf(zscore)
print("p = {}".format( np.round(pvalue,3) ))
```

```
p = 0.126
```

Plot the z-distribution, our z-score test result, and the z_α that corresponds with our 95% confidence interval.

In [358]:

```
# Create z values between -4 and 4 to look at the middle portion of the z-distribution aro
und 0
# Scale our values by the pooled standard deviation (otherwise we'd be in generic z-distri
bution space)
z = np.linspace(-4, 4, num=160) * pooled_sd

# Create the plot
plt.figure(figsize=(10,7))
# Plot the z-distribution here
plt.plot(z, stats.norm.pdf(z, 0, pooled_sd), label='Null PDF: ($\overline{X}_2 - \overline
{X}_1$) = 0')

# Plot a line at our z-alpha value and shade the rejection region
plt.axvline(z_alpha*pooled_sd, color='black', linestyle='-', label='$z_{a}$')
shade = np.linspace(z_alpha*pooled_sd, np.max(z), 10)
plt.fill_between(shade, stats.norm.pdf(shade, 0, pooled_sd) ,  color='k', alpha=0.5, label
='rejection region\nfor alpha={}'.format(np.round(1-conf,2)))


plt.axvline(zscore*pooled_sd, color='red', linestyle='-', label='z-test')
plt.xlabel('($\overline{X}_2 - \overline{X}_1$) [cfs]', fontsize=15)
plt.ylabel('PDF', fontsize=15)
plt.title('Days of Melt')
plt.ticklabel_format(axis='x', style='sci', scilimits=(0,0))
plt.ticklabel_format(axis='y', style='sci', scilimits=(0,0))
plt.ylim(0, 9e-2)
plt.legend(loc='best', fontsize=15);
```



Minimum Discharge

In [ ]:

```
```

In [359]:

```
# Use the cunnane quantile function for before 1975
skookum_before_b = cunnane_quantile(skookum_before, 'maxq')

# Create theoretical normal CDF based on our sample values before 2009
theoretical_cdf_b = stats.norm.cdf(skookum_before_b['maxq'].values,
                                   skookum_before_b['maxq'].mean(),
                                   skookum_before_b['maxq'].std(ddof=1))

# Generate random numbers from a theoretical normal CDF based on our samples before 2009
random_normal_b = np.random.normal(skookum_before_b['maxq'].mean(),
                                   skookum_before_b['maxq'].std(ddof=1),
                                   size=skookum_before_b['maxq'].count())

# Compute the Cunnane plotting position for the random numbers
random_sorted_b, random_quantiles_b  = cunnane_quantile_array(random_normal_b)
```

In [360]:

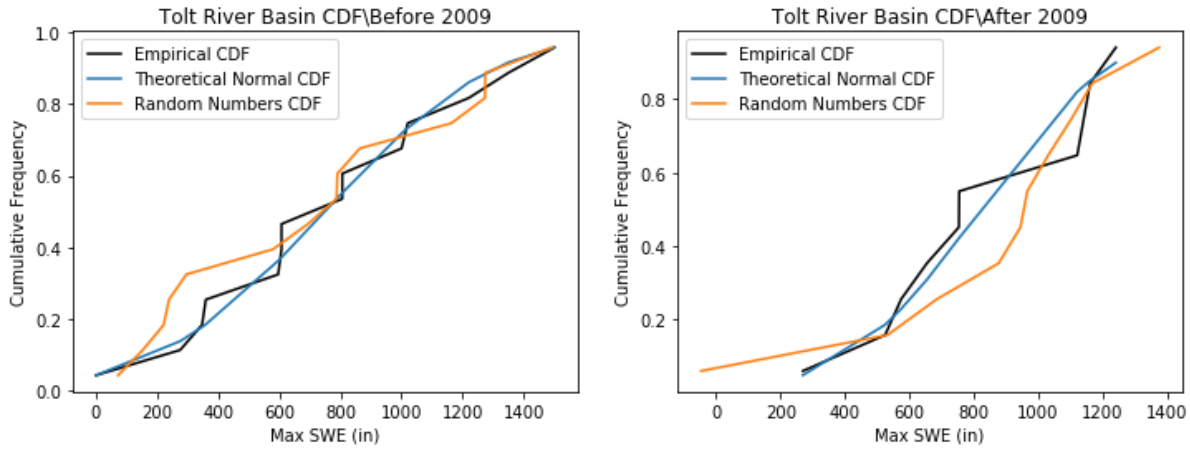
```
# Use the cunnane quantile function for after 1975
skookum_after_a = cunnane_quantile(skookum_after, 'maxq')

# Create theoretical normal CDF based on our sample values before 2009
theoretical_cdf_a = stats.norm.cdf(skookum_after_a['maxq'].values,
                                   skookum_after_a['maxq'].mean(),
                                   skookum_after_a['maxq'].std(ddof=1))

# Generate random numbers from a theoretical normal CDF based on our samples before 2009
random_normal_a = np.random.normal(skookum_after_a['maxq'].mean(),
                                   skookum_after_a['maxq'].std(ddof=1),
                                   size=skookum_after_a['maxq'].count())

# Compute the Cunnane plotting position for the random numbers
random_sorted_a, random_quantiles_a  = cunnane_quantile_array(random_normal_a)
```

In [361]:

```
fig, [ax1, ax2] = plt.subplots(nrows=1, ncols=2, figsize=(12,4))

# Before 2009
# Empirical CDF
ax1.plot(skookum_before_b['maxq'], skookum_before_b['cunnane_plotting_position'], color=
'k', label='Empirical CDF')
# Theorectical Normal CDF
ax1.plot(skookum_before_b['maxq'], theoretical_cdf_b, label='Theoretical Normal CDF')
# Random numbers CDF from a theoretical normal distribution
ax1.plot(random_sorted_b,random_quantiles_b,'-', label='Random Numbers CDF')
# Add legend and labels
ax1.legend()
ax1.set_ylabel('Cumulative Frequency')
ax1.set_xlabel('Max SWE (in)')
ax1.set_title('Tolt River Basin CDF\Before 2009')

# After 2009
# Empirical CDF
ax2.plot(skookum_after_a['maxq'], skookum_after_a['cunnane_plotting_position'], color='k',
label='Empirical CDF')
# Theorectical Normal CDF
ax2.plot(skookum_after_a['maxq'], theoretical_cdf_a, label='Theoretical Normal CDF')
# Random numbers CDF from a theoretical normal distribution
ax2.plot(random_sorted_a, random_quantiles_a,'-', label='Random Numbers CDF')
# Add legend and labels
ax2.legend()
ax2.set_ylabel('Cumulative Frequency')
ax2.set_xlabel('Max SWE (in)')
ax2.set_title('Tolt River Basin CDF\After 2009')
```

Out[361]:

```
Text(0.5, 1.0, 'Tolt River Basin CDF\\After 2009')
```



Does the streamflow data look normally distributed? Maybe try changing the above code to compare the empirical CDFs against theoretical lognormal distributions. (Remember to transform the mean and standard deviations into "log space")

Two-Sample Z-Test

Returning to our question: We are postulating (making a hypothesis) that there was a change in the mean flood statistics after 1975, and we want to test whether this is true. Where do we start?

First we can formally state our null hypothesis, and our alternative hypothesis. We are also told to use a two sample test, and to set α at 5%.

Our **null hypothesis** is that the peak flows of the early period ($\bar{X}_1$) are drawn from the same distribution as the peak flows of the later period ($\bar{X}_2$) (therefore the distributions means of the two time periods are equal):

$H_0: \bar{X}_1 = \bar{X}_2$

Our **alternative hypothesis** is that the mean of the distribution for the later period is greater than that of the early period:

$H_1: \bar{X}_2 > \bar{X}_1$

We can also state these as:

$H_0: \bar{X}_1 - \bar{X}_2 = \mu_0$

$H_1: \bar{X}_1 - \bar{X}_2 < \mu_0$

Where μ_0 is the hypothesized difference between the population means, and in this case $\mu_0 = \mu_1

Note that I have written a "one-sided (https://en.wikipedia.org/wiki/One-_and_two-tailed_tests)" test here because we are testing only for a change in one direction (an increase). We think that either the mean flood increased or it didn't change; we do not think the mean flood decreased:

- This might be chosen because we have some physical reason to think it increased (e.g. higher elevations in the watershed now get rainfall where it used to mostly get snow because of our warming climate).
- Or this might be chosen because we have some practical reason for the test to matter in this particular direction (e.g. we will change flood zoning downstream and/or how we operate a reservoir if the mean flood has increased, but won't make a change if it decreased).

But which test should we use? Is the z-distribution valid?

We are using the z-test (https://en.wikipedia.org/wiki/Z-test), which uses the standard normal distribution. From our work above, we know that our data are likely not neccesarily normally distributed. However, the central limit theorem (https://en.wikipedia.org/wiki/Central_limit_theorem) tells us that, "*If a sample of n values is extracted at random from a population with mean μ and standard deviation σ, and n > 30, then the means of these samples are approximately normally distributed*"

We calculate our z-score as: $\displaystyle Z = \frac{ (\bar{X}_2 - \bar{X}_1) - \mu _{0} } { s_{1,2} }$

Where $s_{1,2}$ is the "pooled standard deviation", s_1, s_2 and n_1, n_2 are the two standard deviations and sample sizes respectively.

$s_{1,2} = \displaystyle\sqrt{ \displaystyle\frac{s^2_1}{n_1} + \displaystyle\frac{s^2_2}{n_2} }$

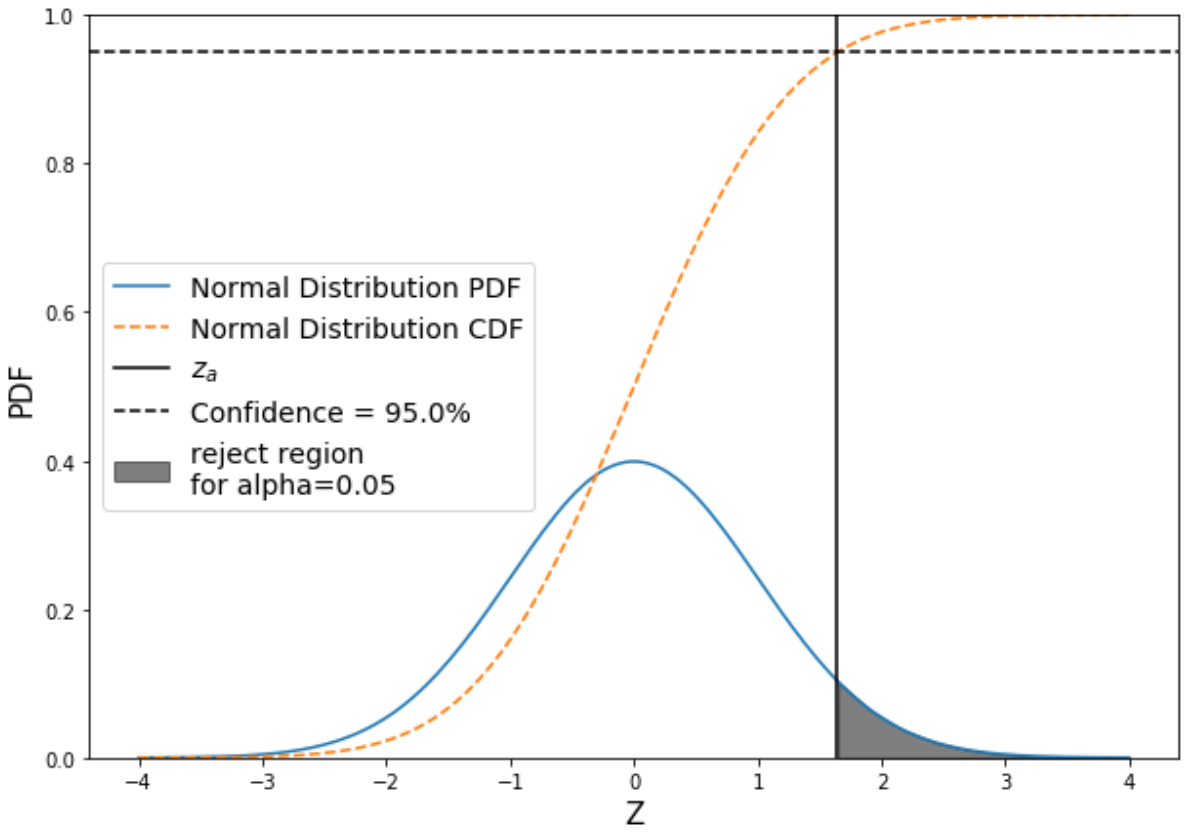
Remember, the means are normally distributed even if the data themselves are not normally distributed.

So what does the "**Null Distribution** look like?

And what do the "rejection regions" look like?

In [362]:

```
fig, ax = plt.subplots(figsize=(10,7))

# Create a null pdf
x = np.linspace(-4, 4, num=160)
ax.plot(x, stats.norm.pdf(x, 0, 1), label='Normal Distribution PDF')

# Plot the null cdf
ax.plot(x, stats.norm.cdf(x, 0, 1), linestyle='--', label='Normal Distribution CDF')

# Plot the region that z_test would have to fall in in order for us to reject the null hyp
othesis
conf = 0.95
z_alpha = stats.norm.ppf(conf)
shade = np.linspace(z_alpha, 4, 10)
ax.fill_between(shade, stats.norm.pdf(shade, 0, 1) ,  color='k', alpha=0.5, label='reject
 region\nfor alpha={}'.format(np.round(1-conf,2)))
# Plot a line at z_alpha
plt.axvline(z_alpha, color='black', label='$z_{a}$')
# Plot a line at our 95% confidence
plt.axhline(conf, color='black', linestyle='--', label='Confidence = {}%'.format(conf*100
))


# Add labels
ax.set_ylim((0,1))
plt.xlabel('Z', fontsize=15)
plt.ylabel('PDF', fontsize=15)
ax.legend(loc='center left', fontsize=14);
```



In [363]:

```
# Check that we have a large enough sample size (n>30)

n = len(skookum_before['maxq'])
print(n)

m = len(skookum_after['maxq'])
print(m)
```

```
14
10
```

Both are larger than 30, so we can go ahead and calculate the z-score for our test:

In [364]:

```
# We want out alpha to be 0.05
alpha = 0.05
# This gives us a confidence of 0.95
conf = 1 - alpha
```

Next, determine which value in the z-distribution corresponds to our 0.95 confidence in the CDF

In [365]:

```
z_alpha = stats.norm.ppf(conf)
print("z_alpha = {}".format(z_alpha))
```

```
z_alpha = 1.6448536269514722
```

Compute the pooled standard deviation, $s_{1,2} = \displaystyle\sqrt{ \displaystyle\frac{s^2_1}{n_1} + \displaystyle\frac{s^2_2}{n_2} }$

In [366]:

```
# Compute the pooled standard deviaiton
pooled_sd = np.sqrt( skookum_before['maxq'].std(ddof=1)**2 / n + skookum_after['maxq'].std
(ddof=1)**2 / m )
```

Finally, compute our z-score as $\displaystyle Z = \frac{ (\bar{X}_2 - \bar{X}_1) - \mu _{0} } { s_{1,2} }$

In [367]:

```
# hypothesizing no change
mu_0 = 0

# compute z-score
zscore = (skookum_after['maxq'].mean() - skookum_before['maxq'].mean() - mu_0)/pooled_sd

print("z-score = {}".format( np.round(zscore,2) ))
```

```
z-score = 0.44
```

We can also compute a p-value from this z-score by looking it up on the standard normal distribution CDF

In [368]:

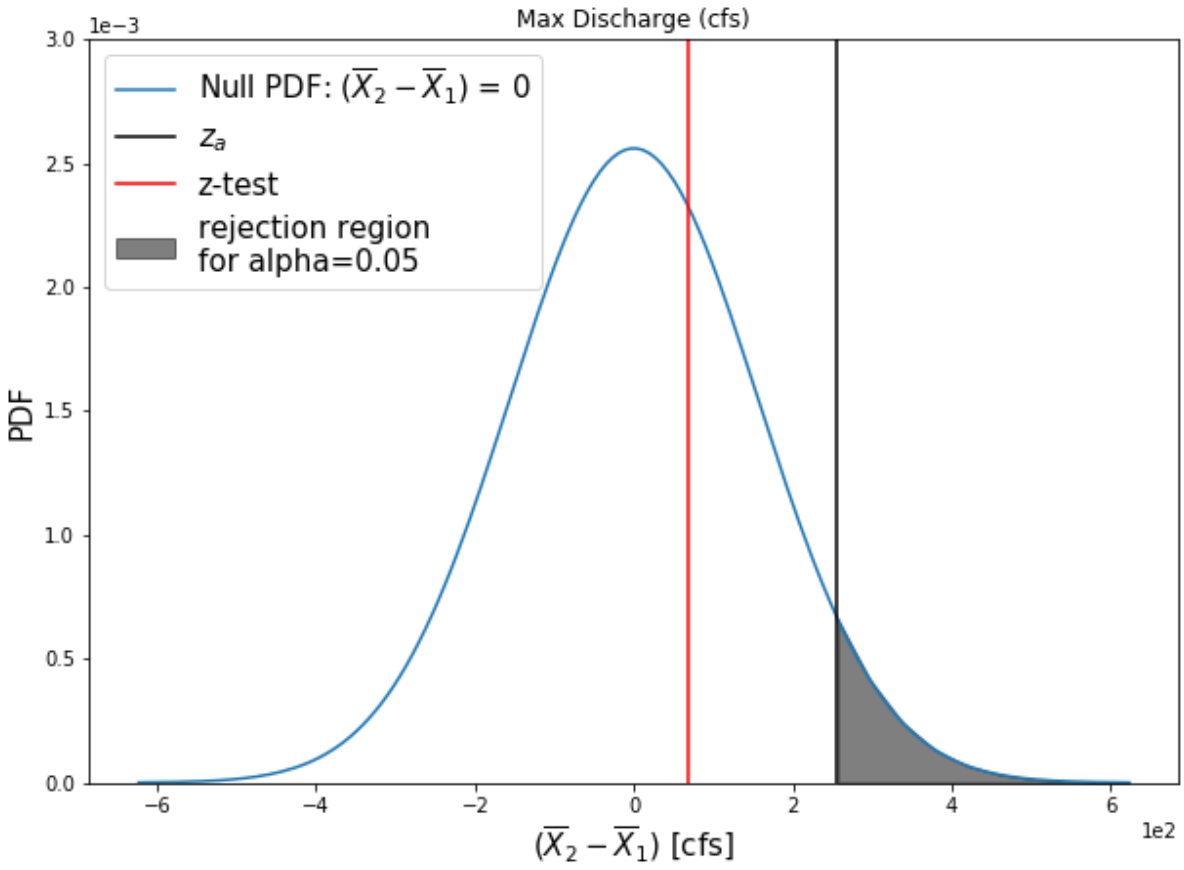
```
pvalue = 1 - stats.norm.cdf(zscore)
print("p = {}".format( np.round(pvalue,3) ))
```

```
p = 0.328
```

Plot the z-distribution, our z-score test result, and the z_α that corresponds with our 95% confidence interval.

In [369]:

```
# Create z values between -4 and 4 to look at the middle portion of the z-distribution aro
und 0
# Scale our values by the pooled standard deviation (otherwise we'd be in generic z-distri
bution space)
z = np.linspace(-4, 4, num=160) * pooled_sd

# Create the plot
plt.figure(figsize=(10,7))
# Plot the z-distribution here
plt.plot(z, stats.norm.pdf(z, 0, pooled_sd), label='Null PDF: ($\overline{X}_2 - \overline
{X}_1$) = 0')

# Plot a line at our z-alpha value and shade the rejection region
plt.axvline(z_alpha*pooled_sd, color='black', linestyle='-', label='$z_{a}$')
shade = np.linspace(z_alpha*pooled_sd, np.max(z), 10)
plt.fill_between(shade, stats.norm.pdf(shade, 0, pooled_sd) ,  color='k', alpha=0.5, label
='rejection region\nfor alpha={}'.format(np.round(1-conf,2)))


plt.axvline(zscore*pooled_sd, color='red', linestyle='-', label='z-test')
plt.xlabel('($\overline{X}_2 - \overline{X}_1$) [cfs]', fontsize=15)
plt.ylabel('PDF', fontsize=15)
plt.title('Max Discharge (cfs)')
plt.ticklabel_format(axis='x', style='sci', scilimits=(0,0))
plt.ticklabel_format(axis='y', style='sci', scilimits=(0,0))
plt.ylim(0, 3e-3)
plt.legend(loc='best', fontsize=15);
```



Day of Year

In [ ]:

In [370]:

```
# Use the cunnane quantile function for before 1975
skookum_before_b = cunnane_quantile(skookum_before, 'dayofyear')

# Create theoretical normal CDF based on our sample values before 2009
theoretical_cdf_b = stats.norm.cdf(skookum_before_b['dayofyear'].values,
                                   skookum_before_b['dayofyear'].mean(),
                                   skookum_before_b['dayofyear'].std(ddof=1))

# Generate random numbers from a theoretical normal CDF based on our samples before 2009
random_normal_b = np.random.normal(skookum_before_b['dayofyear'].mean(),
                                   skookum_before_b['dayofyear'].std(ddof=1),
                                   size=skookum_before_b['dayofyear'].count())

# Compute the Cunnane plotting position for the random numbers
random_sorted_b, random_quantiles_b  = cunnane_quantile_array(random_normal_b)
```

In [371]:

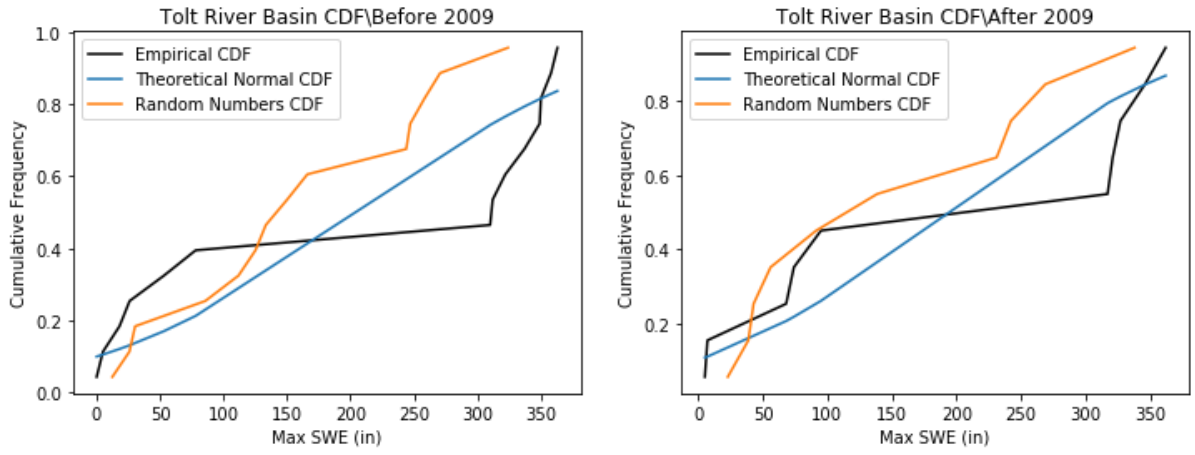
```
# Use the cunnane quantile function for after 1975
skookum_after_a = cunnane_quantile(skookum_after, 'dayofyear')

# Create theoretical normal CDF based on our sample values before 2009
theoretical_cdf_a = stats.norm.cdf(skookum_after_a['dayofyear'].values,
                                   skookum_after_a['dayofyear'].mean(),
                                   skookum_after_a['dayofyear'].std(ddof=1))

# Generate random numbers from a theoretical normal CDF based on our samples before 2009
random_normal_a = np.random.normal(skookum_after_a['dayofyear'].mean(),
                                   skookum_after_a['dayofyear'].std(ddof=1),
                                   size=skookum_after_a['dayofyear'].count())

# Compute the Cunnane plotting position for the random numbers
random_sorted_a, random_quantiles_a  = cunnane_quantile_array(random_normal_a)
```

In [372]:

```
fig, [ax1, ax2] = plt.subplots(nrows=1, ncols=2, figsize=(12,4))

# Before 2009
# Empirical CDF
ax1.plot(skookum_before_b['dayofyear'], skookum_before_b['cunnane_plotting_position'], col
or='k', label='Empirical CDF')
# Theorectical Normal CDF
ax1.plot(skookum_before_b['dayofyear'], theoretical_cdf_b, label='Theoretical Normal CDF')
# Random numbers CDF from a theoretical normal distribution
ax1.plot(random_sorted_b,random_quantiles_b,'-', label='Random Numbers CDF')
# Add legend and labels
ax1.legend()
ax1.set_ylabel('Cumulative Frequency')
ax1.set_xlabel('Max SWE (in)')
ax1.set_title('Tolt River Basin CDF\Before 2009')

# After 2009
# Empirical CDF
ax2.plot(skookum_after_a['dayofyear'], skookum_after_a['cunnane_plotting_position'], color
='k', label='Empirical CDF')
# Theorectical Normal CDF
ax2.plot(skookum_after_a['dayofyear'], theoretical_cdf_a, label='Theoretical Normal CDF')
# Random numbers CDF from a theoretical normal distribution
ax2.plot(random_sorted_a, random_quantiles_a,'-', label='Random Numbers CDF')
# Add legend and labels
ax2.legend()
ax2.set_ylabel('Cumulative Frequency')
ax2.set_xlabel('Max SWE (in)')
ax2.set_title('Tolt River Basin CDF\After 2009')
```

Out[372]:

```
Text(0.5, 1.0, 'Tolt River Basin CDF\\After 2009')
```



Does the streamflow data look normally distributed? Maybe try changing the above code to compare the empirical CDFs against theoretical lognormal distributions. (Remember to transform the mean and standard deviations into "log space")

Two-Sample Z-Test

Returning to our question: We are postulating (making a hypothesis) that there was a change in the mean flood statistics after 1975, and we want to test whether this is true. Where do we start?

First we can formally state our null hypothesis, and our alternative hypothesis. We are also told to use a two sample test, and to set α at 5%.

Our **null hypothesis** is that the peak flows of the early period ($\bar{X}_1$) are drawn from the same distribution as the peak flows of the later period ($\bar{X}_2$) (therefore the distributions means of the two time periods are equal):

$H_0: \bar{X}_1 = \bar{X}_2$

Our **alternative hypothesis** is that the mean of the distribution for the later period is greater than that of the early period:

$H_1: \bar{X}_2 > \bar{X}_1$

We can also state these as:

$H_0: \bar{X}_1 - \bar{X}_2 = \mu_0$

$H_1: \bar{X}_1 - \bar{X}_2 < \mu_0$

Where μ_0 is the hypothesized difference between the population means, and in this case $\mu_0 = \mu_1

Note that I have written a "one-sided (https://en.wikipedia.org/wiki/One-_and_two-tailed_tests)" test here because we are testing only for a change in one direction (an increase). We think that either the mean flood increased or it didn't change; we do not think the mean flood decreased:

- This might be chosen because we have some physical reason to think it increased (e.g. higher elevations in the watershed now get rainfall where it used to mostly get snow because of our warming climate).
- Or this might be chosen because we have some practical reason for the test to matter in this particular direction (e.g. we will change flood zoning downstream and/or how we operate a reservoir if the mean flood has increased, but won't make a change if it decreased).

But which test should we use? Is the z-distribution valid?

We are using the z-test (https://en.wikipedia.org/wiki/Z-test), which uses the standard normal distribution. From our work above, we know that our data are likely not neccesarily normally distributed. However, the central limit theorem (https://en.wikipedia.org/wiki/Central_limit_theorem) tells us that, "*If a sample of n values is extracted at random from a population with mean μ and standard deviation σ, and n > 30, then the means of these samples are approximately normally distributed*"

We calculate our z-score as: $\displaystyle Z = \frac{ (\bar{X}_2 - \bar{X}_1) - \mu _{0} } { s_{1,2} }$

Where $s_{1,2}$ is the "pooled standard deviation", s_1, s_2 and n_1, n_2 are the two standard deviations and sample sizes respectively.

$s_{1,2} = \displaystyle\sqrt{ \displaystyle\frac{s^2_1}{n_1} + \displaystyle\frac{s^2_2}{n_2} }$

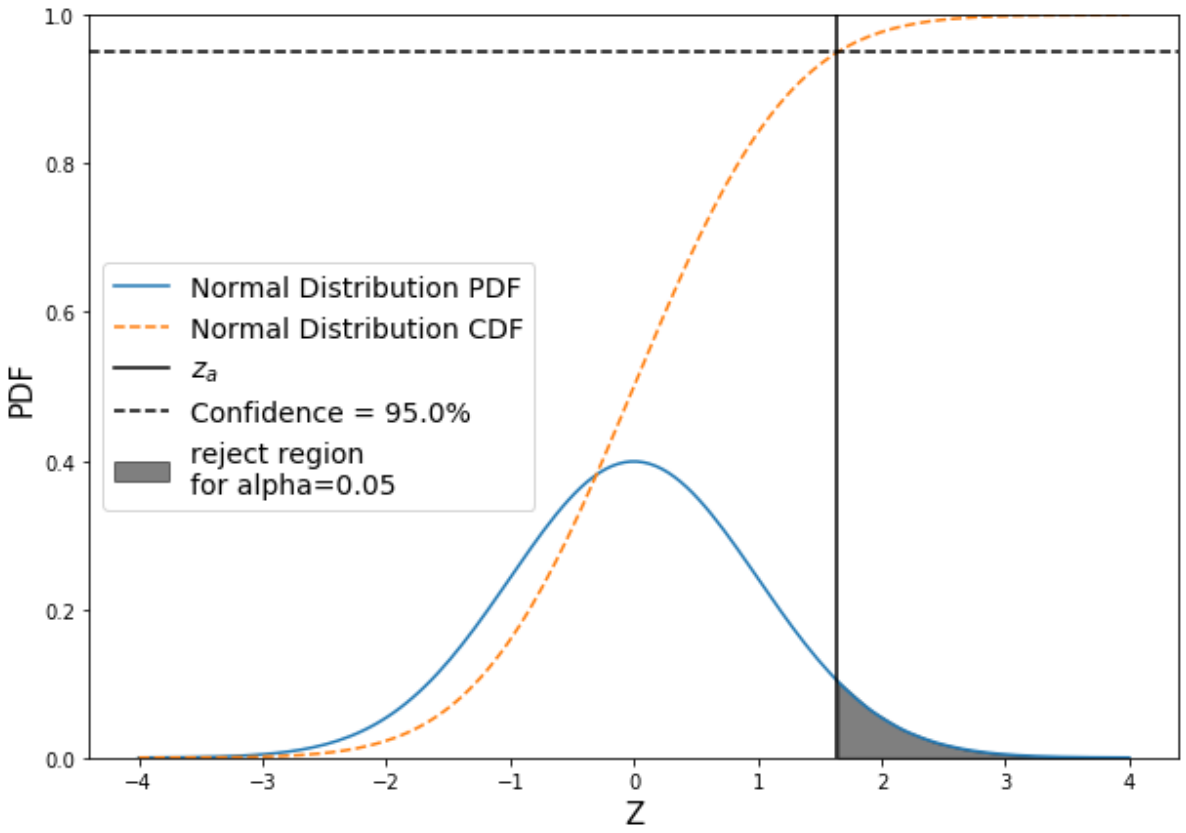
Remember, the means are normally distributed even if the data themselves are not normally distributed.

So what does the "**Null Distribution** look like?

And what do the "rejection regions" look like?

In [373]:

```
fig, ax = plt.subplots(figsize=(10,7))

# Create a null pdf
x = np.linspace(-4, 4, num=160)
ax.plot(x, stats.norm.pdf(x, 0, 1), label='Normal Distribution PDF')

# Plot the null cdf
ax.plot(x, stats.norm.cdf(x, 0, 1), linestyle='--', label='Normal Distribution CDF')

# Plot the region that z_test would have to fall in in order for us to reject the null hyp
othesis
conf = 0.95
z_alpha = stats.norm.ppf(conf)
shade = np.linspace(z_alpha, 4, 10)
ax.fill_between(shade, stats.norm.pdf(shade, 0, 1) ,  color='k', alpha=0.5, label='reject
 region\nfor alpha={}'.format(np.round(1-conf,2)))
# Plot a line at z_alpha
plt.axvline(z_alpha, color='black', label='$z_{a}$')
# Plot a line at our 95% confidence
plt.axhline(conf, color='black', linestyle='--', label='Confidence = {}%'.format(conf*100
))


# Add labels
ax.set_ylim((0,1))
plt.xlabel('Z', fontsize=15)
plt.ylabel('PDF', fontsize=15)
ax.legend(loc='center left', fontsize=14);
```



In [374]:

```
# Check that we have a large enough sample size (n>30)

n = len(skookum_before['dayofyear'])
print(n)

m = len(skookum_after['dayofyear'])
print(m)
```

```
14
10
```

Both are larger than 30, so we can go ahead and calculate the z-score for our test:

In [375]:

```
# We want out alpha to be 0.05
alpha = 0.05
# This gives us a confidence of 0.95
conf = 1 - alpha
```

Next, determine which value in the z-distribution corresponds to our 0.95 confidence in the CDF

In [376]:

```
z_alpha = stats.norm.ppf(conf)
print("z_alpha = {}".format(z_alpha))
```

```
z_alpha = 1.6448536269514722
```

Compute the pooled standard deviation, $s_{1,2} = \displaystyle\sqrt{ \displaystyle\frac{s^2_1}{n_1} + \displaystyle\frac{s^2_2}{n_2} }$

In [377]:

```
# Compute the pooled standard deviaiton
pooled_sd = np.sqrt( skookum_before['dayofyear'].std(ddof=1)**2 / n + skookum_after['dayofyear'].std(ddof=1)**2 / m )
```

Finally, compute our z-score as $\displaystyle Z = \frac{ (\bar{X}_2 - \bar{X}_1) - \mu _{0} } { s_{1,2} }$

In [378]:

```
# hypothesizing no change
mu_0 = 0

# compute z-score
zscore = (skookum_after['dayofyear'].mean() - skookum_before['dayofyear'].mean() - mu_0)/p
ooled_sd

print("z-score = {}".format( np.round(zscore,2) ))
```

```
z-score = -0.21
```

We can also compute a p-value from this z-score by looking it up on the standard normal distribution CDF

In [379]:

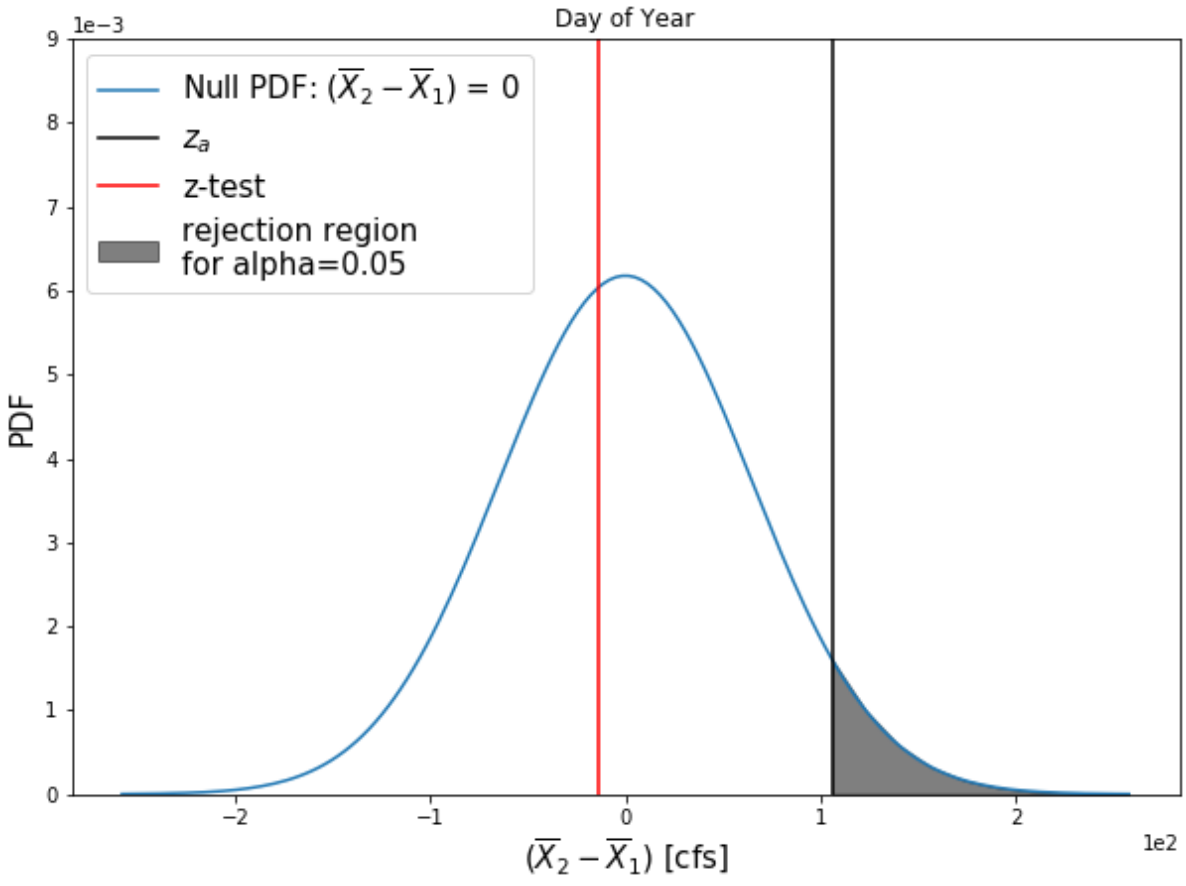
```
pvalue = 1 - stats.norm.cdf(zscore)
print("p = {}".format( np.round(pvalue,3) ))
```

```
p = 0.583
```

Plot the z-distribution, our z-score test result, and the z_α that corresponds with our 95% confidence interval.

In [380]:

```
# Create z values between -4 and 4 to look at the middle portion of the z-distribution aro
und 0
# Scale our values by the pooled standard deviation (otherwise we'd be in generic z-distri
bution space)
z = np.linspace(-4, 4, num=160) * pooled_sd

# Create the plot
plt.figure(figsize=(10,7))
# Plot the z-distribution here
plt.plot(z, stats.norm.pdf(z, 0, pooled_sd), label='Null PDF: ($\overline{X}_2 - \overline
{X}_1$) = 0')

# Plot a line at our z-alpha value and shade the rejection region
plt.axvline(z_alpha*pooled_sd, color='black', linestyle='-', label='$z_{a}$')
shade = np.linspace(z_alpha*pooled_sd, np.max(z), 10)
plt.fill_between(shade, stats.norm.pdf(shade, 0, pooled_sd) ,  color='k', alpha=0.5, label
='rejection region\nfor alpha={}'.format(np.round(1-conf,2)))


plt.axvline(zscore*pooled_sd, color='red', linestyle='-', label='z-test')
plt.xlabel('($\overline{X}_2 - \overline{X}_1$) [cfs]', fontsize=15)
plt.ylabel('PDF', fontsize=15)
plt.title('Day of Year')
plt.ticklabel_format(axis='x', style='sci', scilimits=(0,0))
plt.ticklabel_format(axis='y', style='sci', scilimits=(0,0))
plt.ylim(0, 9e-3)
plt.legend(loc='best', fontsize=15);
```



Melt Rate

In [ ]:

```
```

In [381]:

```
# Use the cunnane quantile function for before 1975
skookum_before_b = cunnane_quantile(skookum_before, 'meltrate')

# Create theoretical normal CDF based on our sample values before 2009
theoretical_cdf_b = stats.norm.cdf(skookum_before_b['meltrate'].values,
                                   skookum_before_b['meltrate'].mean(),
                                   skookum_before_b['meltrate'].std(ddof=1))

# Generate random numbers from a theoretical normal CDF based on our samples before 2009
random_normal_b = np.random.normal(skookum_before_b['meltrate'].mean(),
                                   skookum_before_b['meltrate'].std(ddof=1),
                                   size=skookum_before_b['meltrate'].count())

# Compute the Cunnane plotting position for the random numbers
random_sorted_b, random_quantiles_b  = cunnane_quantile_array(random_normal_b)
```

In [382]:

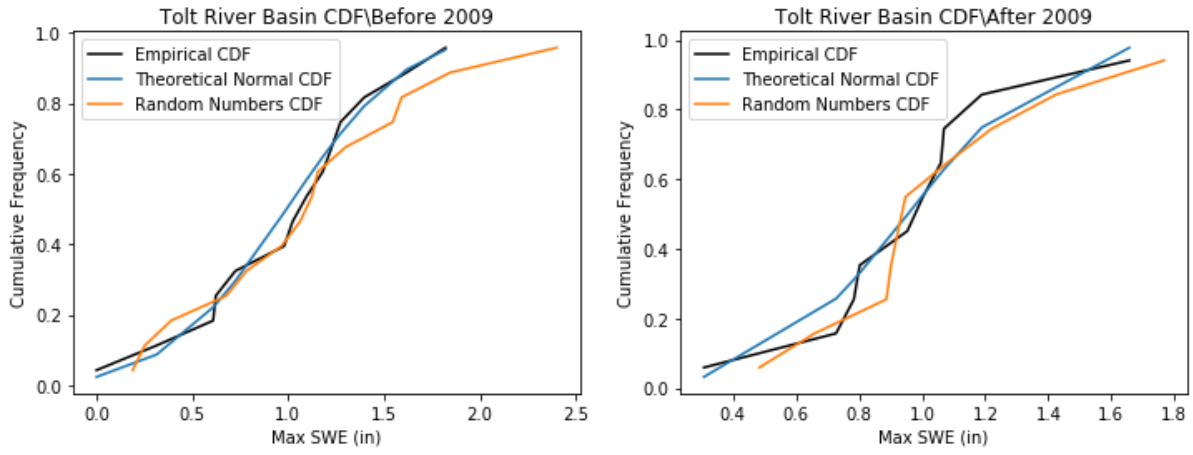
```
# Use the cunnane quantile function for after 1975
skookum_after_a = cunnane_quantile(skookum_after, 'meltrate')

# Create theoretical normal CDF based on our sample values before 2009
theoretical_cdf_a = stats.norm.cdf(skookum_after_a['meltrate'].values,
                                   skookum_after_a['meltrate'].mean(),
                                   skookum_after_a['meltrate'].std(ddof=1))

# Generate random numbers from a theoretical normal CDF based on our samples before 2009
random_normal_a = np.random.normal(skookum_after_a['meltrate'].mean(),
                                   skookum_after_a['meltrate'].std(ddof=1),
                                   size=skookum_after_a['meltrate'].count())

# Compute the Cunnane plotting position for the random numbers
random_sorted_a, random_quantiles_a  = cunnane_quantile_array(random_normal_a)
```

In [383]:

```
fig, [ax1, ax2] = plt.subplots(nrows=1, ncols=2, figsize=(12,4))

# Before 2009
# Empirical CDF
ax1.plot(skookum_before_b['meltrate'], skookum_before_b['cunnane_plotting_position'], colo
r='k', label='Empirical CDF')
# Theorectical Normal CDF
ax1.plot(skookum_before_b['meltrate'], theoretical_cdf_b, label='Theoretical Normal CDF')
# Random numbers CDF from a theoretical normal distribution
ax1.plot(random_sorted_b,random_quantiles_b,'-', label='Random Numbers CDF')
# Add legend and labels
ax1.legend()
ax1.set_ylabel('Cumulative Frequency')
ax1.set_xlabel('Max SWE (in)')
ax1.set_title('Tolt River Basin CDF\Before 2009')

# After 2009
# Empirical CDF
ax2.plot(skookum_after_a['meltrate'], skookum_after_a['cunnane_plotting_position'], color=
'k', label='Empirical CDF')
# Theorectical Normal CDF
ax2.plot(skookum_after_a['meltrate'], theoretical_cdf_a, label='Theoretical Normal CDF')
# Random numbers CDF from a theoretical normal distribution
ax2.plot(random_sorted_a, random_quantiles_a,'-', label='Random Numbers CDF')
# Add legend and labels
ax2.legend()
ax2.set_ylabel('Cumulative Frequency')
ax2.set_xlabel('Max SWE (in)')
ax2.set_title('Tolt River Basin CDF\After 2009')
```

Out[383]:

```
Text(0.5, 1.0, 'Tolt River Basin CDF\\After 2009')
```



Does the streamflow data look normally distributed? Maybe try changing the above code to compare the empirical CDFs against theoretical lognormal distributions. (Remember to transform the mean and standard deviations into "log space")

Two-Sample Z-Test

Returning to our question: We are postulating (making a hypothesis) that there was a change in the mean flood statistics after 1975, and we want to test whether this is true. Where do we start?

First we can formally state our null hypothesis, and our alternative hypothesis. We are also told to use a two sample test, and to set α at 5%.

Our **null hypothesis** is that the peak flows of the early period ($\bar{X}_1$) are drawn from the same distribution as the peak flows of the later period ($\bar{X}_2$) (therefore the distributions means of the two time periods are equal):

$H_0: \bar{X}_1 = \bar{X}_2$

Our **alternative hypothesis** is that the mean of the distribution for the later period is greater than that of the early period:

$H_1: \bar{X}_2 > \bar{X}_1$

We can also state these as:

$H_0: \bar{X}_1 - \bar{X}_2 = \mu_0$

$H_1: \bar{X}_1 - \bar{X}_2 < \mu_0$

Where μ_0 is the hypothesized difference between the population means, and in this case $\mu_0 = \mu_1

Note that I have written a "one-sided (https://en.wikipedia.org/wiki/One-_and_two-tailed_tests)" test here because we are testing only for a change in one direction (an increase). We think that either the mean flood increased or it didn't change; we do not think the mean flood decreased:

- This might be chosen because we have some physical reason to think it increased (e.g. higher elevations in the watershed now get rainfall where it used to mostly get snow because of our warming climate).
- Or this might be chosen because we have some practical reason for the test to matter in this particular direction (e.g. we will change flood zoning downstream and/or how we operate a reservoir if the mean flood has increased, but won't make a change if it decreased).

But which test should we use? Is the z-distribution valid?

We are using the z-test (https://en.wikipedia.org/wiki/Z-test), which uses the standard normal distribution. From our work above, we know that our data are likely not neccesarily normally distributed. However, the central limit theorem (https://en.wikipedia.org/wiki/Central_limit_theorem) tells us that, "*If a sample of n values is extracted at random from a population with mean μ and standard deviation σ, and n > 30, then the means of these samples are approximately normally distributed*"

We calculate our z-score as: $\displaystyle Z = \frac{ (\bar{X}_2 - \bar{X}_1) - \mu _{0} } { s_{1,2} }$

Where $s_{1,2}$ is the "pooled standard deviation", s_1, s_2 and n_1, n_2 are the two standard deviations and sample sizes respectively.

$s_{1,2} = \displaystyle\sqrt{ \displaystyle\frac{s^2_1}{n_1} + \displaystyle\frac{s^2_2}{n_2} }$

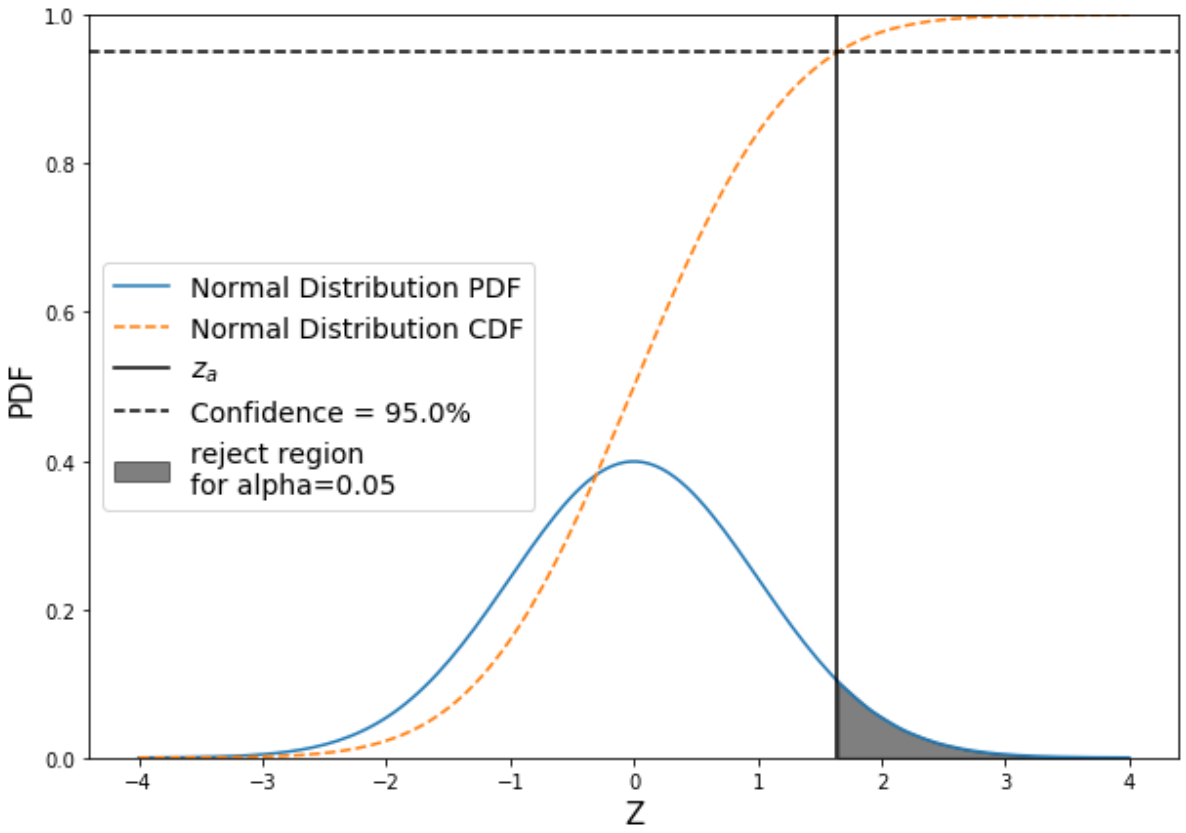
Remember, the means are normally distributed even if the data themselves are not normally distributed.

So what does the "**Null Distribution** look like?

And what do the "rejection regions" look like?

In [384]:

```
fig, ax = plt.subplots(figsize=(10,7))

# Create a null pdf
x = np.linspace(-4, 4, num=160)
ax.plot(x, stats.norm.pdf(x, 0, 1), label='Normal Distribution PDF')

# Plot the null cdf
ax.plot(x, stats.norm.cdf(x, 0, 1), linestyle='--', label='Normal Distribution CDF')

# Plot the region that z_test would have to fall in in order for us to reject the null hyp
othesis
conf = 0.95
z_alpha = stats.norm.ppf(conf)
shade = np.linspace(z_alpha, 4, 10)
ax.fill_between(shade, stats.norm.pdf(shade, 0, 1) ,  color='k', alpha=0.5, label='reject
 region\nfor alpha={}'.format(np.round(1-conf,2)))
# Plot a line at z_alpha
plt.axvline(z_alpha, color='black', label='$z_{a}$')
# Plot a line at our 95% confidence
plt.axhline(conf, color='black', linestyle='--', label='Confidence = {}%'.format(conf*100
))


# Add labels
ax.set_ylim((0,1))
plt.xlabel('Z', fontsize=15)
plt.ylabel('PDF', fontsize=15)
ax.legend(loc='center left', fontsize=14);
```



In [385]:

```
# Check that we have a large enough sample size (n>30)

n = len(skookum_before['meltrate'])
print(n)

m = len(skookum_after['meltrate'])
print(m)
```

```
14
10
```

Both are larger than 30, so we can go ahead and calculate the z-score for our test:

In [386]:

```
# We want out alpha to be 0.05
alpha = 0.05
# This gives us a confidence of 0.95
conf = 1 - alpha
```

Next, determine which value in the z-distribution corresponds to our 0.95 confidence in the CDF

In [387]:

```
z_alpha = stats.norm.ppf(conf)
print("z_alpha = {}".format(z_alpha))
```

```
z_alpha = 1.6448536269514722
```

Compute the pooled standard deviation, $s_{1,2} = \displaystyle\sqrt{ \displaystyle\frac{s^2_1}{n_1} + \displaystyle\frac{s^2_2}{n_2} }$

In [388]:

```
# Compute the pooled standard deviaiton
pooled_sd = np.sqrt( skookum_before['meltrate'].std(ddof=1)**2 / n + skookum_after['meltrate'].std(ddof=1)**2 / m )
```

Finally, compute our z-score as $\displaystyle Z = \frac{ (\bar{X}_2 - \bar{X}_1) - \mu _{0} } { s_{1,2} }$

In [389]:

```
# hypothesizing no change
mu_0 = 0

# compute z-score
zscore = (skookum_after['meltrate'].mean() - skookum_before['meltrate'].mean() - mu_0)/poo
led_sd

print("z-score = {}".format( np.round(zscore,2) ))
```

```
z-score = -0.21
```

We can also compute a p-value from this z-score by looking it up on the standard normal distribution CDF

In [390]:

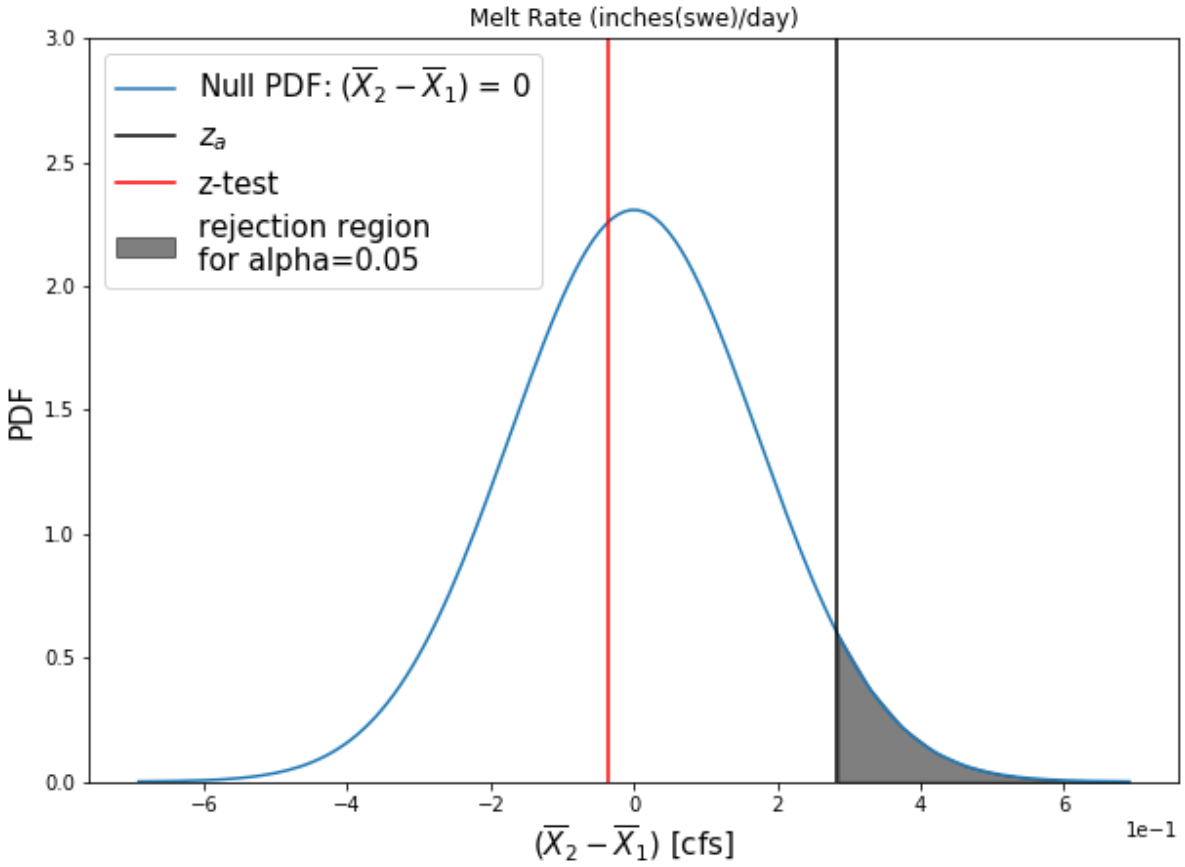
```
pvalue = 1 - stats.norm.cdf(zscore)
print("p = {}".format( np.round(pvalue,3) ))
```

```
p = 0.584
```

Plot the z-distribution, our z-score test result, and the z_α that corresponds with our 95% confidence interval.

In [399]:

```
# Create z values between -4 and 4 to look at the middle portion of the z-distribution aro
und 0
# Scale our values by the pooled standard deviation (otherwise we'd be in generic z-distri
bution space)
z = np.linspace(-4, 4, num=160) * pooled_sd

# Create the plot
plt.figure(figsize=(10,7))
# Plot the z-distribution here
plt.plot(z, stats.norm.pdf(z, 0, pooled_sd), label='Null PDF: ($\overline{X}_2 - \overline
{X}_1$) = 0')

# Plot a line at our z-alpha value and shade the rejection region
plt.axvline(z_alpha*pooled_sd, color='black', linestyle='-', label='$z_{a}$')
shade = np.linspace(z_alpha*pooled_sd, np.max(z), 10)
plt.fill_between(shade, stats.norm.pdf(shade, 0, pooled_sd) ,  color='k', alpha=0.5, label
='rejection region\nfor alpha={}'.format(np.round(1-conf,2)))


plt.axvline(zscore*pooled_sd, color='red', linestyle='-', label='z-test')
plt.xlabel('($\overline{X}_2 - \overline{X}_1$) [cfs]', fontsize=15)
plt.ylabel('PDF', fontsize=15)
plt.title('Melt Rate (inches(swe)/day)')
plt.ticklabel_format(axis='x', style='sci', scilimits=(0,0))
plt.ticklabel_format(axis='y', style='sci', scilimits=(0,0))
plt.ylim(0, 3e-0)
plt.legend(loc='best', fontsize=15);
```



In [ ]: