Exercise 3

In this exercise, you will analyse a dataset obtained from the London transport system (TfL). The data is in a filled called `tfl_readership.csv` (comma-separated-values format). We will load and view the data using a package called `pandas`.

In [ ]:
```
# If you are running this on Google Colab, uncomment and run the following lines; ot
# from google.colab import drive
# drive.mount('/content/drive')
```

In [1]:
```
import math
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [2]:
```
# Load data
df_tfl = pd.read_csv('tfl_ridership.csv')
# If running on Google Colab change path to '/content/drive/MyDrive/IB-Data-Science/

df_tfl.head(13)
```

Out[2]:

	Year	Period	Start	End	Days	Bus cash (000s)	Bus Oyster PAYG (000s)	Bus Contactless (000s)	Bus One Day Bus Pass (000s)	Bus Day Travelcard (000s)	...	Tu Contactl (00
0	2000/01	P 01	01 Apr '00	29 Apr '00	29d	884	0	0	210	231	...	
1	2000/01	P 02	30 Apr '00	27 May '00	28d	949	0	0	214	205	...	
2	2000/01	P 03	28 May '00	24 Jun '00	28d	945	0	0	209	221	...	
3	2000/01	P 04	25 Jun '00	22 Jul '00	28d	981	0	0	216	241	...	
4	2000/01	P 05	23 Jul '00	19 Aug '00	28d	958	0	0	225	248	...	
5	2000/01	P 06	20 Aug '00	16 Sep '00	28d	984	0	0	243	236	...	
6	2000/01	P 07	17 Sep '00	14 Oct '00	28d	1001	0	0	205	216	...	
7	2000/01	P 08	15 Oct '00	11 Nov '00	28d	979	0	0	199	221	...	

	Year	Period	Start	End	Days	Bus cash (000s)	Bus Oyster PAYG (000s)	Bus Contactless (000s)	Bus One Day Bus Pass (000s)	Bus Day Travelcard (000s)	...	Tu Contactl (00
8	2000/01	P 09	12 Nov '00	09 Dec '00	28d	971	0	0	184	212	...	
9	2000/01	P 10	10 Dec '00	06 Jan '01	28d	912	0	0	192	211	...	
10	2000/01	P 11	07 Jan '01	03 Feb '01	28d	943	0	0	193	186	...	
11	2000/01	P 12	04 Feb '01	03 Mar '01	28d	975	0	0	194	210	...	
12	2000/01	P 13	04 Mar '01	31 Mar '01	28d	974	0	0	186	204	...	

13 rows × 26 columns

Each row of our data frame represents the average daily ridership over a 28/29 day period for various types of transport and tickets (bus, tube etc.). We have used the `.head()` command to display the top 13 rows of the data frame (corresponding to one year). Focusing on the "Tube Total" column, notice the dip in ridership in row 9 (presumably due to Christmas/New Year's), and also the slight dip during the summer (rows 4,5).

In [3]:
```
#df_tfl.sample(3)  #random sample of 3 rows
df_tfl.tail(3)  #last 3 rows
```

Out[3]:

	Year	Period	Start	End	Days	Bus cash (000s)	Bus Oyster PAYG (000s)	Bus Contactless (000s)	Bus One Day Bus Pass (000s)	Bus Day Travelcard (000s)	...	Contac (0
242	2018/19	P 09	11 Nov '18	08 Dec '18	28d	0	1110	1089	0	41	...	
243	2018/19	P 10	09 Dec '18	05 Jan '19	28d	0	1001	949	0	38	...	
244	2018/19	P 11	06 Jan '19	02 Feb '19	28d	0	1036	1075	0	30	...	

3 rows × 26 columns

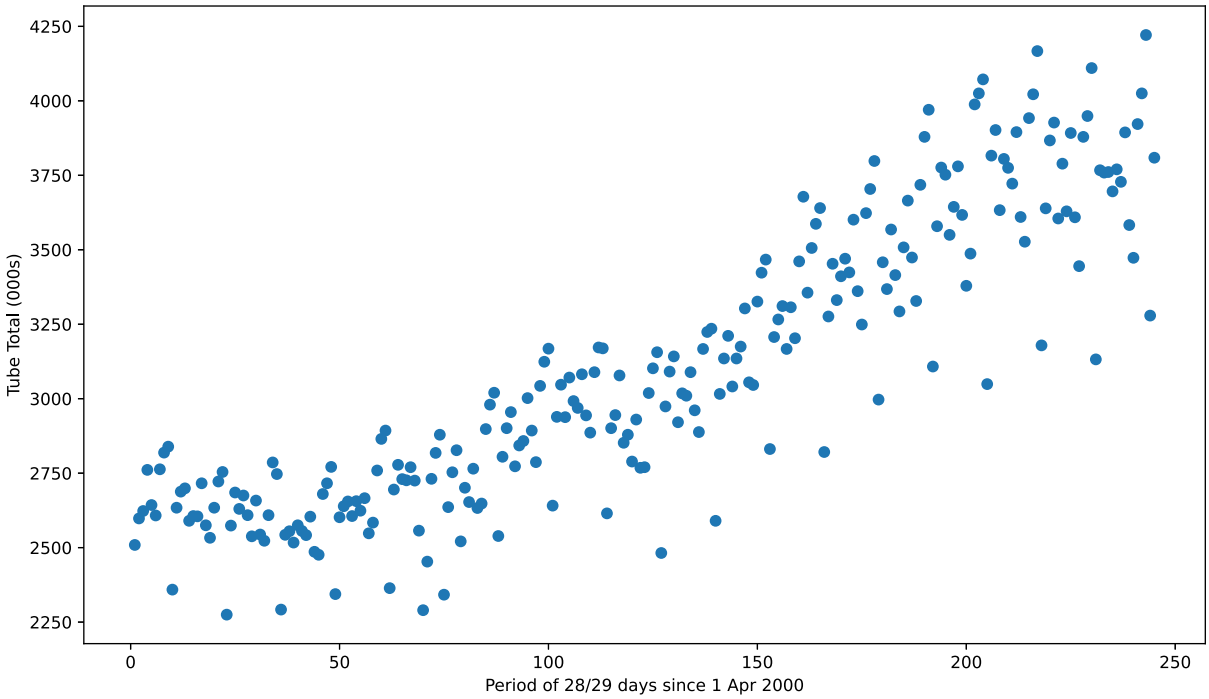
The dataframe contains $N = 245$ counting periods (of 28/29 days each) from 1 April 2000 to 2 Feb 2019. We now define a numpy array consisting of the values in the ' Tube Total (000s)' column:

In [4]:
```
yvals = np.array(df_tfl['Tube Total (000s)'])
N = np.size(yvals)
xvals = np.linspace(1,N,N) #an array containing the values 1,2....,N
```

We now have a time series consisting of points (x_i, y_i), for $i = 1, \ldots, N$, where y_i is the average daily tube rideship in counting period $x_i = i$.

2a) Plot the data in a scatterplot

In [5]:
```
plt.rcParams['figure.figsize'] = [12, 7]
plt.scatter(xvals, yvals)
plt.xlabel("Period of 28/29 days since 1 Apr 2000")
plt.ylabel("Tube Total (000s)")
plt.show()
```



2b) Fit a linear model $f(x) = \beta_0 + \beta_1 x$ to the data

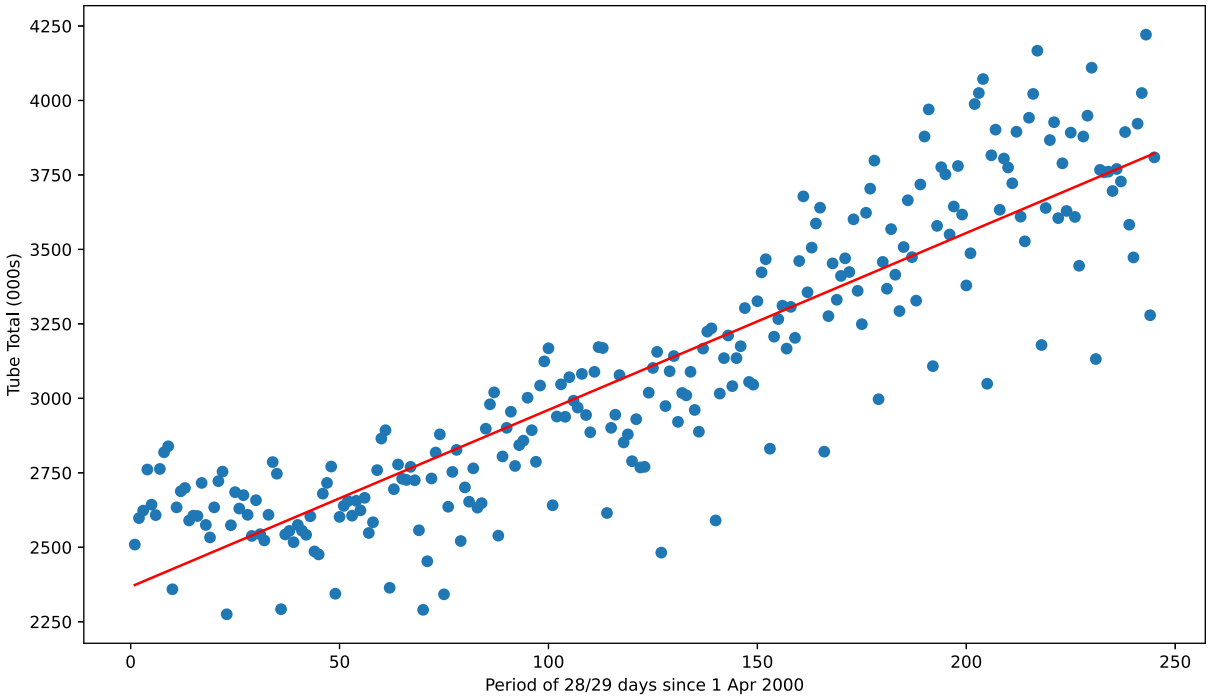
- Print the values of the regression coefficients β_0, β_1 determined using least-squares.
- Plot the fitted model and the scatterplot on the same plot.
- Compute and print the **MSE** and the R^2 coefficient for the fitted model.

All numerical outputs should be displayed to three decimal places.

In [6]:
```
β1, β0 = np.polyfit(xvals, yvals,1)
print(f"β\N{SUBSCRIPT ZERO}: {β0:.3f}\nβ\N{SUBSCRIPT ONE}: {β1:.3f}")

yfit = β0 + β1 * xvals
SSE = sum((yfit - yvals)**2)

print(f"MSE: {SSE / N:.3f}\nR\N{SUPERSCRIPT TWO}: {1 - SSE/(N * np.var(yvals)):3g}")

plt.plot(xvals, yfit, "r")
```

```
plt.scatter(xvals, yvals)
plt.xlabel("Period of 28/29 days since 1 Apr 2000")
plt.ylabel("Tube Total (000s)")
plt.show()
```

```
β₀: 2367.382
β₁: 5.939
MSE: 45323.636
R²: 0.795611
```

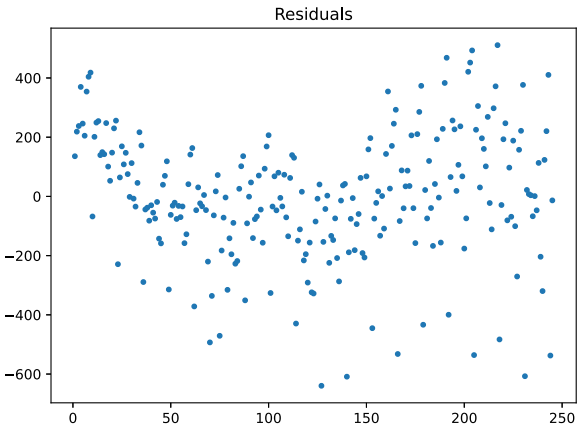


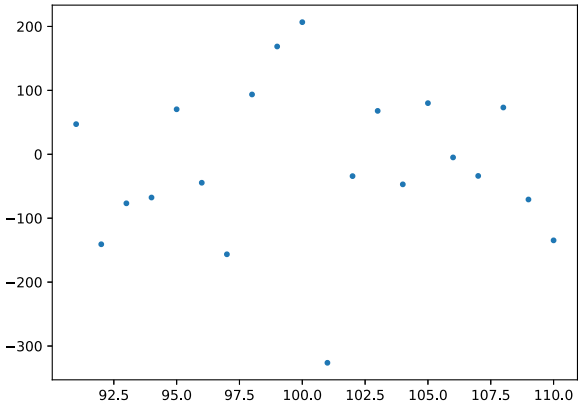
2c) Plotting the residuals

- Plot the residuals on a scatterplot
- Also plot the residuals over a short duration and comment on whether you can discern any periodic components.

In [7]:
```
resid = yvals - yfit

fig, axs = plt.subplots(ncols=2, figsize=(15,5))
axs[0].scatter(xvals, resid, s=10)
axs[0].set_title("Residuals")

# Plot residuals for zoomed in time period
start = 90
stop = 110
axs[1].scatter(xvals[start:stop], resid[start:stop], s=10)
axs[1].set_title("")
```

Out[7]:
```
Text(0.5, 1.0, '')
```



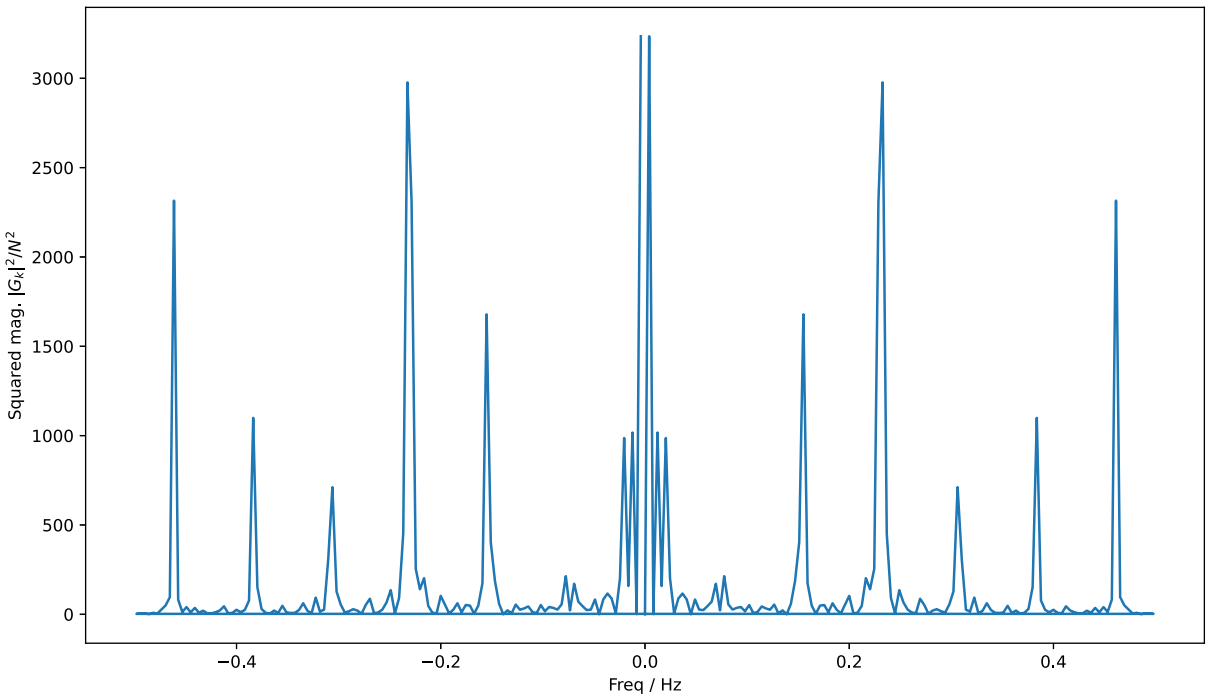


No obvious periodic component

2d) Periodogram

- Compute and plot the peridogram of the residuals. (Recall that the periodogram is the squared-magnitude of the DFT coefficients.)
- Identify the indices/frequencies for which the periogram value exceeds **50%** of the maximum.

In [8]:
```
# Your code to compute and plot the histogram
T = 1 # Time period of xvals is 1 unit, representing 1 period of 28/29 days

periodogram = np.abs(np.fft.fft(resid, N) / N) ** 2
freq = np.fft.fftfreq(resid.shape[-1])
plt.plot(freq, periodogram)
plt.xlabel("Freq / Hz")
plt.ylabel("Squared mag. $|G_k|^2/N^2$")
plt.show()
```



In [9]:
```
# Your code to identify the indices for which the periodogram value exceeds 50% of t
np.set_printoptions(formatter={'float': lambda x: "{0:0.3f}".format(x)})

mask = tuple([periodogram > 0.5*np.max(periodogram)]) # list of True or False depend
```

```
main_freq = np.array([f for f in freq[mask] if f > 0])
print(f"Poitive Frequencies for which peridogram value exceeds 50% of the maximum\n{
```

```
Poitive Frequencies for which peridogram value exceeds 50% of the maximum
[0.004 0.155 0.229 0.233 0.461]
```

2e) To the residuals, fit a model of the form

$$\beta_{1s}\sin(\omega_1 x) + \beta_{1c}\cos(\omega_1 x) + \beta_{2s}\sin(\omega_2 x) + \beta_{2c}\cos(\omega_2 x) + \ldots + \beta_{Ks}\sin(\omega_K x) + \beta_{Kc}\cos(\omega_I$$

The frequencies $\omega_1, \ldots, \omega_K$ in the model are those corresponding to the indices identified in Part 2c. (Hint: Each of the sines and cosines will correspond to one column in your X-matrix.)

- Print the values of the regression coefficients obtained using least-squares.
- Compute and print the final **MSE** and R^2 coefficient. Comment on the improvement over the linear fit.

All numerical outputs should be displayed to three decimal places.

In [10]:

```
# Your code here
def trigfit(x, y, freq):
    """Fit sum of sinusoids for y
    freq: Hz
    """
    X = np.column_stack(
                np.concatenate(
                    (
                        [np.sin(2 * np.pi * f * x) for f in freq],
                        [np.cos(2 * np.pi * f * x) for f in freq]
                    )
                )
    )

    # print(X[0])
    # print(X.shape)

    beta = np.linalg.inv(X.T.dot(X)).dot(X.T).dot(y)    # Least squares
    fit = X.dot(beta)                                   # Fit

    return beta, fit, y-fit

beta, trig_fit, resid2 = trigfit(xvals, resid, main_freq)

beta_sin = beta[:len(main_freq)]
beta_cos = beta[len(main_freq):]
print(f"Regression coefficients\nfreq: {main_freq}\nsin: {beta_sin}\ncos: {beta_cos}

SSE = sum(resid2 ** 2)
print(f"MSE: {SSE / N:.3f}")
print(f"R\N{SUPERSCRIPT TWO}: {1 - SSE / (N * np.var(yvals)):.3f}")
```

```
Regression coefficients
freq: [0.004 0.155 0.229 0.233 0.461]
sin: [-51.253 61.628 -15.581 81.659 32.472]
cos: [101.556 -54.006 -94.797 72.381 90.589]
MSE: 20297.501
R²: 0.908
```

2f) The combined fit

- Plot the combined fit together with a scatterplot of the data
- Compute and print the final **MSE** and R^2 coefficient. Comment on the improvement over the linear fit.

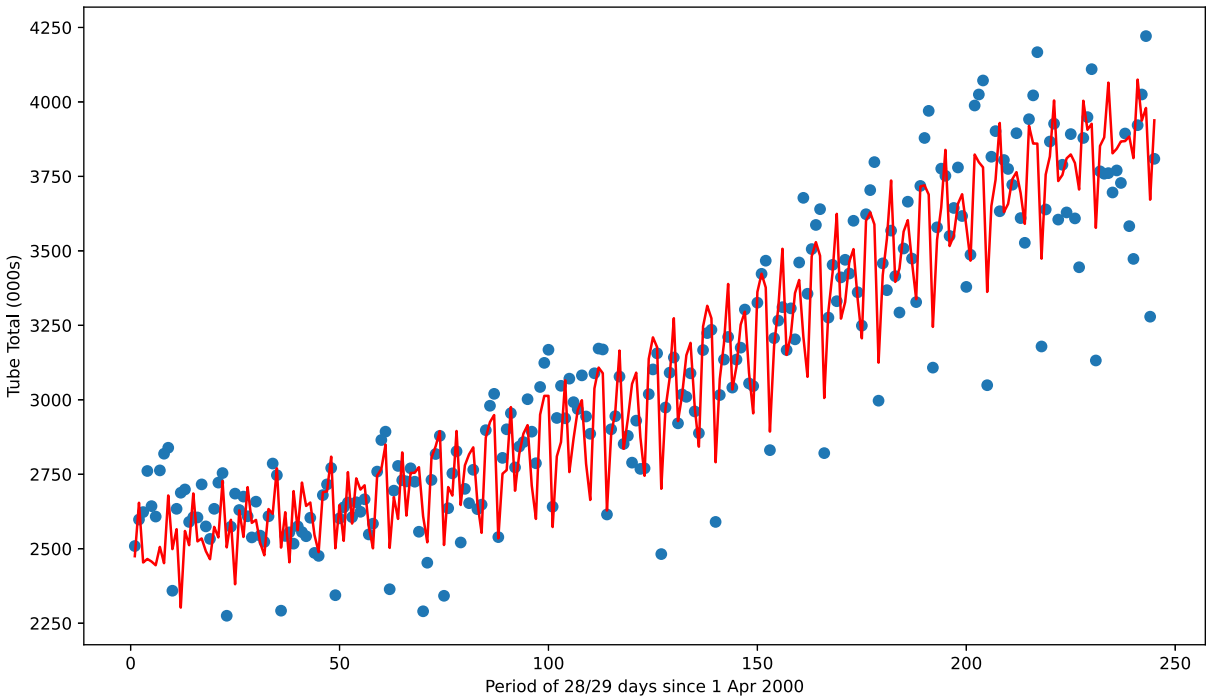
The combined fit, which corresponds to the full model

$$f(x) = \beta_0 + \beta_1 x + \beta_{s1}\sin(\omega_1 x) + \beta_{c1}\cos(\omega_1 x) + \ldots + \beta_{sk}\sin(\omega_k x) + \beta_{ck}\cos(\omega_k x),$$

can be obtained by adding the fits in parts 2b) and 2e).

In [13]:
```
fit = yfit + trig_fit

SSE = sum((yvals - fit) ** 2)
print(f"MSE: {SSE / N:.3f}")
print(f"R\N{SUPERSCRIPT TWO}: {1 - SSE / (N * np.var(yvals)):.3f}")

plt.plot(xvals, fit, "r")
plt.scatter(xvals, yvals)
plt.xlabel("Period of 28/29 days since 1 Apr 2000")
plt.ylabel("Tube Total (000s)")
plt.show()
```

```
MSE: 20297.501
R²: 0.908
```



MSE decreased by factor of 2

R^2 has improved from 0.8 to 0.9