เวกเตอร์และเมทริกซ์

เวกเตอร์ (Vector) และเมทริกซ์ (Matrix) มีความสำคัญในการศึกษาคณิตศาสตร์และการคำนวณ อีกทั้งยังเป็นพื้นฐานที่จำเป็นของการเขียนกราฟิก ผู้ศึกษาต้องคุ้นเคยกับการใช้สัญลักษณ์เวกเตอร์และเมทริกซ์รวมไปถึงคุณสมบัติอื่น ๆ ความรู้พื้นฐานที่จำเป็นนี้เองทำให้เราต้องเข้าใจในคุณสมบัติและต้องจดจำกระบวนการหลาย ๆ อย่าง เช่น การบวก ลบ หรือการคูณ 2 เมทริกซ์เข้าด้วยกัน การหาเมทริกซ์ผกผันจากเมทริกซ์ที่กำหนดมาให้ การต้องจดจำกระบวนการต่าง ๆ เหล่านี้ทั้ง ๆ ที่ยังไม่ทราบว่าจะนำไปใช้ประโยชน์ต่อไปอย่างไรได้โดยชัดเจน ทำให้ผู้ศึกษาจำนวนไม่น้อยเกิดความเบื่อหน่าย อีกทั้งจะลืมกระบวนการเหล่านี้ไปเกือบจนหมดหลังจากที่จบการศึกษาไปแล้ว

เวกเตอร์และเมทริกซ์ยังเป็นหัวใจที่สำคัญสำหรับการประดิษฐ์ซอฟต์แวร์ขนาดใหญ่เพื่อใช้แก้ปัญหาทางวิทยาศาสตร์และวิศวกรรมศาสตร์ในปัจจุบัน ทั้งนี้ก็เพราะเหตุผลหลักที่ว่าเวกเตอร์และเมทริกซ์ช่วยทำให้การจัดการภายในซอฟต์แวร์เหล่านี้เป็นไปได้อย่างมีประสิทธิภาพ ซอฟต์แวร์ขนาดใหญ่ที่มีราคาสูงล้วนตั้งอยู่บนฐานองค์ความรู้ของเวกเตอร์และเมทริกซ์ ซอฟต์แวร์ไพธอนบรรจุคำสั่งที่สามารถจัดการกับเวกเตอร์และเมทริกซ์ได้โดยสะดวก ทำให้เราเกิดความเข้าใจในกระบวนการและผลลัพธ์ที่เกิดขึ้นได้อย่างชัดเจน ดังนั้นในบทนี้เราจะมาทบทวนการจัดการกับเวกเตอร์และเมทริกซ์ด้วยการใช้ตัวอย่างที่ง่าย ๆ โดยเราต้องตระหนักอยู่เสมอว่ากระบวนการเช่นเดียวกันนี้สามารถนำไปประยุกต์ใช้กับเวกเตอร์และเมทริกซ์ที่มีขนาดใหญ่ ซึ่งใช้ในงานทางปฏิบัติได้เช่นกัน

แพ็กเกจ NumPy สำหรับการสร้างอาร์เรย์

อาร์เรย์ (Array) ใน 1 มิติ หมายถึง เมทริกซ์ที่มีเพียงแถวเดียว ซึ่งอาจเป็นเวกเตอร์แบบแถวนอน (Row Vector) เช่น

$$\begin{bmatrix} 3 & 5 & 1 & 9 \end{bmatrix}$$

หรือเวกเตอร์แบบแถวตั้ง (Column Vector) เช่น

$$\begin{bmatrix} 8 \\ 3 \end{bmatrix}$$

ส่วนอาร์เรย์ใน 2 มิติ หมายถึง เมทริกซ์ที่มีทั้งแถวนอนและแถวตั้ง เช่น

$$\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

อาร์เรย์ที่อยู่ในรูปแบบของเวกเตอร์และเมทริกซ์นี้ได้ถูกนำมาประยุกต์ใช้ในศาสตร์ของการคำนวณกันอย่างกว้างขวาง

แพ็กเกจ NumPy บรรจุคำสั่งและฟังก์ชันพื้นฐานเพื่อช่วยอำนวยความสะดวกในการดำเนินการที่เกี่ยวข้องกับเวกเตอร์และเมทริกซ์ได้เป็นอย่างดี คำสั่งและฟังก์ชันพื้นฐานในแพ็กเกจนี้มีเป็นจำนวนมาก

ซึ่งสามารถแสดงได้ด้วยคำสั่ง

In [1]:

```
import numpy
dir(numpy)
```

Out[1]:

```
['ALLOW_THREADS',
 'AxisError',
 'BUFSIZE',
 'Bytes0',
 'CLIP',
 'ComplexWarning',
 'DataSource',
 'Datetime64',
 'ERR_CALL',
 'ERR_DEFAULT',
 'ERR_IGNORE',
 'ERR_LOG',
 'ERR_PRINT',
 'ERR_RAISE',
 'ERR_WARN',
 'FLOATING_POINT_SUPPORT',
 'FPE_DIVIDEBYZERO',
 'FPE_INVALID',
 'FPE_OVERFLOW',
 'FPE_UNDERFLOW',
 'False_',
 'Inf',
 'Infinity',
 'MAXDIMS',
 'MAY_SHARE_BOUNDS',
 'MAY_SHARE_EXACT',
 'MachAr',
 'ModuleDeprecationWarning',
 'NAN',
 'NINF',
 'NZERO',
 'NaN',
 'PINF',
 'PZERO',
 'RAISE',
 'RankWarning',
 'SHIFT_DIVIDEBYZERO',
 'SHIFT_INVALID',
 'SHIFT_OVERFLOW',
 'SHIFT_UNDERFLOW',
 'ScalarType',
 'Str0',
 'Tester',
 'TooHardError',
 'True_',
 'UFUNC_BUFSIZE_DEFAULT',
 'UFUNC_PYVALS_NAME',
 'Uint64',
 'VisibleDeprecationWarning',
 'WRAP',
 '_NoValue',
 '_UFUNC_API',
 '__NUMPY_SETUP__',
 '__all__',
```

```
'__builtins__',
'__cached__',
'__config__',
'__deprecated_attrs__',
'__dir__',
'__doc__',
'__expired_functions__',
'__file__',
'__getattr__',
'__git_revision__',
'__loader__',
'__name__',
'__package__',
'__path__',
'__spec__',
'__version__',
'_add_newdoc_ufunc',
'_distributor_init',
'_financial_names',
'_globals',
'_mat',
'_pytesttester',
'abs',
'absolute',
'add',
'add_docstring',
'add_newdoc',
'add_newdoc_ufunc',
'alen',
'all',
'allclose',
'alltrue',
'amax',
'amin',
'angle',
'any',
'append',
'apply_along_axis',
'apply_over_axes',
'arange',
'arccos',
'arccosh',
'arcsin',
'arcsinh',
'arctan',
'arctan2',
'arctanh',
'argmax',
'argmin',
'argpartition',
'argsort',
'argwhere',
'around',
'array',
'array2string',
'array_equal',
'array_equiv',
'array_repr',
'array_split',
'array_str',
'asanyarray',
```

```
'asarray',
'asarray_chkfinite',
'ascontiguousarray',
'asfarray',
'asfortranarray',
'asmatrix',
'asscalar',
'atleast_1d',
'atleast_2d',
'atleast_3d',
'average',
'bartlett',
'base_repr',
'binary_repr',
'bincount',
'bitwise_and',
'bitwise_not',
'bitwise_or',
'bitwise_xor',
'blackman',
'block',
'bmat',
'bool8',
'bool_',
'broadcast',
'broadcast_arrays',
'broadcast_shapes',
'broadcast_to',
'busday_count',
'busday_offset',
'busdaycalendar',
'byte',
'byte_bounds',
'bytes0',
'bytes_',
'c_',
'can_cast',
'cast',
'cbrt',
'cdouble',
'ceil',
'cfloat',
'char',
'character',
'chararray',
'choose',
'clip',
'clongdouble',
'clongfloat',
'column_stack',
'common_type',
'compare_chararrays',
'compat',
'complex128',
'complex256',
'complex64',
'complex_',
'complexfloating',
'compress',
'concatenate',
'conj',
```

```
 'conjugate',
 'convolve',
 'copy',
 'copysign',
 'copyto',
 'core',
 'corrcoef',
 'correlate',
 'cos',
 'cosh',
 'count_nonzero',
 'cov',
 'cross',
 'csingle',
 'ctypeslib',
 'cumprod',
 'cumproduct',
 'cumsum',
 'datetime64',
 'datetime_as_string',
 'datetime_data',
 'deg2rad',
 'degrees',
 'delete',
 'deprecate',
 'deprecate_with_doc',
 'diag',
 'diag_indices',
 'diag_indices_from',
 'diagflat',
 'diagonal',
 'diff',
 'digitize',
 'disp',
 'divide',
 'divmod',
 'dot',
 'double',
 'dsplit',
 'dstack',
 'dtype',
 'e',
 'ediff1d',
 'einsum',
 'einsum_path',
 'emath',
 'empty',
 'empty_like',
 'equal',
 'error_message',
 'errstate',
 'euler_gamma',
 'exp',
 'exp2',
 'expand_dims',
 'expm1',
 'extract',
 'eye',
 'fabs',
 'fastCopyAndTranspose',
 'fft',
```

```
            'fill_diagonal',
            'find_common_type',
            'finfo',
            'fix',
            'flatiter',
            'flatnonzero',
            'flexible',
            'flip',
            'fliplr',
            'flipud',
            'float128',
            'float16',
            'float32',
            'float64',
            'float_',
            'float_power',
            'floating',
            'floor',
            'floor_divide',
            'fmax',
            'fmin',
            'fmod',
            'format_float_positional',
            'format_float_scientific',
            'format_parser',
            'frexp',
            'frombuffer',
            'fromfile',
            'fromfunction',
            'fromiter',
            'frompyfunc',
            'fromregex',
            'fromstring',
            'full',
            'full_like',
            'gcd',
            'generic',
            'genfromtxt',
            'geomspace',
            'get_array_wrap',
            'get_include',
            'get_printoptions',
            'getbufsize',
            'geterr',
            'geterrcall',
            'geterrobj',
            'gradient',
            'greater',
            'greater_equal',
            'half',
            'hamming',
            'hanning',
            'heaviside',
            'histogram',
            'histogram2d',
            'histogram_bin_edges',
            'histogramdd',
            'hsplit',
            'hstack',
            'hypot',
            'i0',
```

```
'identity',
'iinfo',
'imag',
'in1d',
'index_exp',
'indices',
'inexact',
'inf',
'info',
'infty',
'inner',
'insert',
'int0',
'int16',
'int32',
'int64',
'int8',
'int_',
'intc',
'integer',
'interp',
'intersect1d',
'intp',
'invert',
'is_busday',
'isclose',
'iscomplex',
'iscomplexobj',
'isfinite',
'isfortran',
'isin',
'isinf',
'isnan',
'isnat',
'isneginf',
'isposinf',
'isreal',
'isrealobj',
'isscalar',
'issctype',
'issubclass_',
'issubdtype',
'issubsctype',
'iterable',
'ix_',
'kaiser',
'kron',
'lcm',
'ldexp',
'left_shift',
'less',
'less_equal',
'lexsort',
'lib',
'linalg',
'linspace',
'little_endian',
'load',
'loads',
'loadtxt',
'log',
```

```
 'log10',
 'log1p',
 'log2',
 'logaddexp',
 'logaddexp2',
 'logical_and',
 'logical_not',
 'logical_or',
 'logical_xor',
 'logspace',
 'longcomplex',
 'longdouble',
 'longfloat',
 'longlong',
 'lookfor',
 'ma',
 'mafromtxt',
 'mask_indices',
 'mat',
 'math',
 'matmul',
 'matrix',
 'matrixlib',
 'max',
 'maximum',
 'maximum_sctype',
 'may_share_memory',
 'mean',
 'median',
 'memmap',
 'meshgrid',
 'mgrid',
 'min',
 'min_scalar_type',
 'minimum',
 'mintypecode',
 'mod',
 'modf',
 'moveaxis',
 'msort',
 'multiply',
 'nan',
 'nan_to_num',
 'nanargmax',
 'nanargmin',
 'nancumprod',
 'nancumsum',
 'nanmax',
 'nanmean',
 'nanmedian',
 'nanmin',
 'nanpercentile',
 'nanprod',
 'nanquantile',
 'nanstd',
 'nansum',
 'nanvar',
 'nbytes',
 'ndarray',
 'ndenumerate',
 'ndfromtxt',
```

```
 'ndim',
 'ndindex',
 'nditer',
 'negative',
 'nested_iters',
 'newaxis',
 'nextafter',
 'nonzero',
 'not_equal',
 'numarray',
 'number',
 'obj2sctype',
 'object0',
 'object_',
 'ogrid',
 'oldnumeric',
 'ones',
 'ones_like',
 'os',
 'outer',
 'packbits',
 'pad',
 'partition',
 'percentile',
 'pi',
 'piecewise',
 'place',
 'poly',
 'poly1d',
 'polyadd',
 'polyder',
 'polydiv',
 'polyfit',
 'polyint',
 'polymul',
 'polynomial',
 'polysub',
 'polyval',
 'positive',
 'power',
 'printoptions',
 'prod',
 'product',
 'promote_types',
 'ptp',
 'put',
 'put_along_axis',
 'putmask',
 'quantile',
 'r_',
 'rad2deg',
 'radians',
 'random',
 'ravel',
 'ravel_multi_index',
 'real',
 'real_if_close',
 'rec',
 'recarray',
 'recfromcsv',
 'recfromtxt',
```

```
 'reciprocal',
 'record',
 'remainder',
 'repeat',
 'require',
 'reshape',
 'resize',
 'result_type',
 'right_shift',
 'rint',
 'roll',
 'rollaxis',
 'roots',
 'rot90',
 'round',
 'round_',
 'row_stack',
 's_',
 'safe_eval',
 'save',
 'savetxt',
 'savez',
 'savez_compressed',
 'sctype2char',
 'sctypeDict',
 'sctypes',
 'searchsorted',
 'select',
 'set_numeric_ops',
 'set_printoptions',
 'set_string_function',
 'setbufsize',
 'setdiff1d',
 'seterr',
 'seterrcall',
 'seterrobj',
 'setxor1d',
 'shape',
 'shares_memory',
 'short',
 'show_config',
 'sign',
 'signbit',
 'signedinteger',
 'sin',
 'sinc',
 'single',
 'singlecomplex',
 'sinh',
 'size',
 'sometrue',
 'sort',
 'sort_complex',
 'source',
 'spacing',
 'split',
 'sqrt',
 'square',
 'squeeze',
 'stack',
 'std',
```

```
 'str0',
 'str_',
 'string_',
 'subtract',
 'sum',
 'swapaxes',
 'sys',
 'take',
 'take_along_axis',
 'tan',
 'tanh',
 'tensordot',
 'test',
 'testing',
 'tile',
 'timedelta64',
 'trace',
 'tracemalloc_domain',
 'transpose',
 'trapz',
 'tri',
 'tril',
 'tril_indices',
 'tril_indices_from',
 'trim_zeros',
 'triu',
 'triu_indices',
 'triu_indices_from',
 'true_divide',
 'trunc',
 'typeDict',
 'typecodes',
 'typename',
 'ubyte',
 'ufunc',
 'uint',
 'uint0',
 'uint16',
 'uint32',
 'uint64',
 'uint8',
 'uintc',
 'uintp',
 'ulonglong',
 'unicode_',
 'union1d',
 'unique',
 'unpackbits',
 'unravel_index',
 'unsignedinteger',
 'unwrap',
 'use_hugepage',
 'ushort',
 'vander',
 'var',
 'vdot',
 'vectorize',
 'version',
 'void',
 'void0',
 'vsplit',
```

```
 'vstack',
 'w',
 'warnings',
 'where',
 'who',
 'zeros',
 'zeros_like']
```

ในบทนี้เราจะศึกษาเฉพาะคำสั่งและฟังก์ชันที่สำคัญ ๆ ซึ่งจำเป็นต้องใช้ในคณิตศาสตร์การคำนวณเท่านั้น

การสร้างเวกเตอร์และเมทริกซ์

เริ่มจากการสร้างเวกเตอร์แบบแถวนอน (Row Vector) ขนาด (1×2) คือ 1 row และ 2 columns เช่น

In [2]:
```
import numpy as np
a = np.array([1, 2])
a
```

Out[2]:
```
array([1, 2])
```

และเวกเตอร์แบบแถวตั้ง (Column Vector) ขนาด (2×1) คือ 2 rows และ 1 column เช่น

In [3]:
```
b = np.array([[3], [4]])
b
```

Out[3]:
```
array([[3],
       [4]])
```

รวมทั้งเมทริกซ์ใน 2 มิติ ขนาด (2×2) คือ 2 rows และ 2 columns เช่น

In [4]:
```
c = np.array([[5, 6], [7, 8]])
c
```

Out[4]:
```
array([[5, 6],
       [7, 8]])
```

แพ็กเกจ NumPy บรรจุหลายคำสั่งเพื่อสร้างเมทริกซ์ที่มีลักษณะจำเพาะได้โดยสะดวก เช่น ต้องการสร้างเมทริกซ์ขนาด (3×3) ที่เอลิเมนต์ (Element) ทุกตัวมีค่าเป็นศูนย์

In [5]:
```
z = np.zeros((3,3))
z
```

Out[5]:
```
array([[0., 0., 0.],
       [0., 0., 0.],
       [0., 0., 0.]])
```

หรือเมทริกซ์ขนาด (2×3) ที่เอลิเมนต์ทุกตัวมีค่าเป็นหนึ่ง

In [6]:
```
o = np.ones((2, 3))
o
```

Out[6]:
```
array([[1., 1., 1.],
       [1., 1., 1.]])
```

In [4]:
```
MF = 5 * np.ones((3,3))
MF
```

Out[4]:
```
array([[5., 5., 5.],
       [5., 5., 5.],
       [5., 5., 5.]])
```

และเมทริกซ์เอกลักษณ์ (Identity Matrix) ขนาด (3×3)

In [7]:
```
i = np.eye(3)
i
```

Out[7]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

รวมทั้งการทำทรานสโพสเมทริกซ์ (Matrix Transpose) เช่น ต้องการหาค่าทรานสโพสของเวกเตอร์แนวนอน a เวกเตอร์แนวตั้ง b และเมทริกซ์ c ก่อนหน้านี้

In [8]:
```
a = np.array ([1, 2])
np.transpose([a])
```

Out[8]:
```
array([[1],
       [2]])
```

In [9]:
```
b = np.array([[3], [4]])
np.transpose(b)
```

Out[9]:
```
array([[3, 4]])
```

In [10]:
```
c = np.array([[5, 6], [7, 8]])
np.transpose(c)
```

Out[10]:
```
array([[5, 7],
       [6, 8]])
```

สิ่งสำคัญสิ่งหนึ่งที่ต้องตระหนักอยู่เสมอในขณะใช้ซอฟต์แวร์ไพธอนก็คือ วิธีการใช้หมายเลขของดัชนี (Index) เพื่อกำกับตำแหน่งของเอลิเมนต์ในเมทริกซ์ ซอฟต์แวร์ไพธอนกำหนดหมายเลขดัชนีเริ่มต้นจากหมายเลขศูนย์เสมอทั้งในแถวนอนและแถวตั้งในขณะที่ซอฟต์แวร์อื่น ๆ (เช่น MATLAB, Mathematica และ Fortran) เริ่มต้นจากดัชนีหมายเลขหนึ่ง ยกตัวอย่างเช่น เรามีเมทริกซ์ d ขนาด (3×3) ซึ่งบรรจุตัวเลข ดังนี้

In [24]:
```
d = np.array([[1., 2., 3.], [4., 5., 6.], [7., 8., 9.]])
d
```

Out[24]:
```
array([[1., 2., 3.],
       [4., 5., 6.],
       [7., 8., 9.]])
```

ในเมทริกซ์ d ข้างต้น เอลิเมนต์บนซ้ายสุดซึ่งมีค่าเท่ากับ 1 เป็นเอลิเมนต์ที่มีค่าดัชนีระบุตำแหน่งเป็น (0,0) คืออยู่ที่ row หมายเลขศูนย์และ column หมายเลขศูนย์ ไม่ใช่ที่ตำแหน่ง (1,1) หรือ row หมายเลขหนึ่ง และ column หมายเลขหนึ่ง ดังที่คุ้นเคยกันในซอฟต์แวร์อื่น ๆ หมายเลขตำแหน่งของดัชนีในซอฟต์แวร์ไพธอนเช่นนี้ยืนยันได้จาก

In [12]:
```
print(d[0][0])
```

```
1
```

ในทำนองเดียวกัน หมายเลข 8 ในเมทริกซ์ d นี้มีค่าดัชนีระบุตำแหน่งเป็น (2,1)

In [13]:
```
print(d[2][1])
```

```
8
```

ในขณะที่เอลิเมนต์ที่มีดัชนีระบุตำแหน่งเป็น (3,3) ไม่ปรากฏอยู่ในเมทริกซ์ d นี้

In [14]:
```
print(d[3][3])
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-14-6ab3f3f80e24> in <module>
----> 1 print(d[3][3])

IndexError: index 3 is out of bounds for axis 0 with size 3
```

In [13]:
```
d
```

Out[13]:
```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

In [15]:
```
d[0] = np.array([100, 200, 300])
```

In [17]:
```
d[0] = (1/100) * d[0]
```

In [42]:
```
d = np.array([[1., 2., 3.], [4., 5., 6.], [7., 8., 9.]])
d
```

Out[42]:
```
array([[1., 2., 3.],
       [4., 5., 6.],
       [7., 8., 9.]])
```

In [43]:
```
d[1] = (1/5) * d[1]
d
```

Out[43]:
```
array([[1. , 2. , 3. ],
       [0.8, 1. , 1.2],
       [7. , 8. , 9. ]])
```

In [44]:
```
d[0] = -2 * d[1] + d[0]
d
```

Out[44]:
```
array([[-0.6,  0. ,  0.6],
       [ 0.8,  1. ,  1.2],
       [ 7. ,  8. ,  9. ]])
```

In [39]:
```
d[2] = -8 * d[1] + d[2]
d
```

Out[39]:
```
array([[-0.6,  0. ,  0.6],
       [ 0.8,  1. ,  1.2],
       [ 0.6,  0. , -0.6]])
```

ดังนั้น การประดิษฐ์โปรแกรมซึ่งต้องเกี่ยวข้องกับหมายเลข row และ column ในเมทริกซ์ จึงต้องกระทำอย่างระมัดระวังในขณะใช้ซอฟต์แวร์ไพธอน

การดำเนินการทางคณิตศาสตร์ของเวกเตอร์และเมทริกซ์

ในหัวข้อนี้เราจะศึกษาการนำเวกเตอร์และเมทริกซ์ที่ได้สร้างขึ้นมาดำเนินการทางคณิตศาสตร์ระหว่างกัน เช่น การบวก การลบ การคูณ การหาค่าดีเทอร์มิแนนต์และค่าเมทริกซ์ผกผัน โดยจะเริ่มต้นจากการดำเนินการทางคณิตศาสตร์สำหรับเวกเตอร์ก่อน สมมติว่าเรามีเวกเตอร์ 2 เวกเตอร์ เช่น

In [54]:
```
u = np.array([1, 5, 6])
v = np.array([3, 7, 2])
print('u = ', u)
print('v = ', v)
```

```
u =  [1 5 6]
v =  [3 7 2]
```

In [63]:
```
b = np.array([[1,2,3], [4,5,6]])
print(b)
np.shape(b)
```

```
[[1 2 3]
 [4 5 6]]
```

Out[63]:
```
(2, 3)
```

In [66]:
```
m = np.array([1,2,3])
n = np.array([ [1], [2], [3] ])
print('m = ', m)
print('n = ', n)
print(np.shape(m))
print(np.shape(n))
```

```
m =  [1 2 3]
n =  [[1]
 [2]
 [3]]
(3,)
(3, 1)
```

การบวกและลบระหว่างเวกเตอร์นั้นทำได้โดยตรง ดังนี้

In [55]:
```
w = u + v
w
```

Out[55]:
```
array([ 4, 12,  8])
```

In [56]:
```
w = u - v
w
```

Out[56]:
```
array([-2, -2,  4])
```

แต่ถ้าต้องการหาผลลัพธ์จากการดอท (Dot Product) หรือหาผลลัพธ์จากการครอส (Cross Product) ก็สามารถใช้คำสั่งดังต่อไปนี้ได้ตามลำดับ

In [18]:
```
w = np.dot(u, v)
w
```

Out[18]:
```
50
```

In [19]:
```
w = np.cross(u, v)
w
```

Out[19]:
```
array([-32,  16,  -8])
```

In [20]:
```
w = np.cross(v, u)
w
```

Out[20]:
```
array([ 32, -16,   8])
```

เราสามารถดำเนินการทางคณิตศาสตร์ระหว่างปริมาณสเกลาร์ (Scalar) กับเวกเตอร์ได้ทั้งการบวก ลบ คูณ และหาร โดยปริมาณสเกลาร์จะถูกเข้าไปบวก ลบ คูณ และหารกับแต่ละเอลิเมนต์ของเวกเตอร์นั้น ดังตัวอย่างต่อไปนี้

In [21]:
```
u + 2
```

Out[21]:
```
array([3, 7, 8])
```

In [67]:
```
u + 2 * np.ones( np.shape(u) )
```

Out[67]:
```
array([3., 7., 8.])
```

In [22]:
```
u - 2
```

Out[22]:
```
array([-1,  3,  4])
```

In [23]:
```
u * 2
```

Out[23]:
```
array([ 2, 10, 12])
```

In [24]:
```
u / 2
```

Out[24]:
```
array([0.5, 2.5, 3. ])
```

เราสามารถทำการบวก ลบ และคูณเมทริกซ์เข้าด้วยกัน โดยการบวกและลบนั้น เมทริกซ์ทั้งสองต้องมีขนาดที่เท่ากัน ยกตัวอย่างเช่น ถ้าเรามีเมทริกซ์ 3 เมทริกซ์ดังนี้

In [68]:
```
M = np.array([[3, 4, 6], [8, 7, 2]])
```

In [69]:
```
N = np.array([[1, 0, 5], [9, 6, 3]])
```

In [70]:
```
P = np.array([[2, 3], [1, 5], [4, 9]])
```

โดยเมทริกซ์ M และ N มีขนาด (2×3) ส่วนเมทริกซ์ P มีขนาด (3×2) เราสามารถดำเนินการบวกหรือลบระหว่างเมทริกซ์ M กับ N ได้ แต่ไม่สามารถดำเนินการดังกล่าวระหว่างเมทริกซ์ M กับ P ได้ เนื่องจากมีขนาดไม่เท่ากัน ดังตัวอย่างต่อไปนี้

In [71]:
```
M + N
```

Out[71]:
```
array([[ 4,  4, 11],
       [17, 13,  5]])
```

In [72]:
```
N - M
```

Out[72]:
```
array([[-2, -4, -1],
       [ 1, -1,  1]])
```

In [73]:
```
M + P
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-73-ced00130ac11> in <module>
----> 1 M + P

ValueError: operands could not be broadcast together with shapes (2,3) (3,2
)
```

In [76]:
```
M
```

Out[76]:
```
array([[3, 4, 6],
       [8, 7, 2]])
```

In [77]:
```
P
```

Out[77]:
```
array([[2, 3],
       [1, 5],
       [4, 9]])
```

In [79]:
```
P * M
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-79-9afc396df1b9> in <module>
----> 1 P * M

ValueError: operands could not be broadcast together with shapes (3,2) (2,3
)
```

สำหรับการคูณเมทริกซ์นั้น จำนวน Column ของเมทริกซ์ตัวตั้งต้องเท่ากับจำนวน Low ของเมทริกซ์ตัวคูณ ดังนั้นเราสามารถคูณเมทริกซ์ P ด้วยเมทริกซ์ M ได้

In [31]:
```
P = np.matrix(P)
```

In [32]:
```
M = np.matrix(M)
```

In [33]:
```
P * M
```

Out[33]:
```
matrix([[30, 29, 18],
        [43, 39, 16],
        [84, 79, 42]])
```

ในการกลับกัน เราสามารถคูณเมทริกซ์ M ด้วยเมทริกซ์ P ได้ เพียงแต่ผลลัพธ์ที่ได้นั้นต่างกัน

In [34]:
```
M * P
```

Out[34]:
```
matrix([[34, 83],
        [31, 77]])
```

ส่วนการหาค่าดีเทอร์มิแนนต์ (Determinant) ของเมทริกซ์ เมทริกซ์นั้นต้องเป็นเมทริกซ์จัตุรัส เราสามารถใช้คำสั่งดังนี้เพื่อหาค่าดีเทอร์มิแนนต์

In [35]:
```
F = np.array([[1, 4, 7], [2, 5, 6], [3, 4, 8]])
```

In [36]:
```
np.linalg.det(F)
```

Out[36]:
```
-25.000000000000007
```

แต่ถ้าต้องหาค่าเมทริกซ์ผกผัน (Inverse) เราก็ใช้คำสั่งว่า

In [37]:
```
np.linalg.inv(F)
```

Out[37]:
```
array([[-0.64,  0.16,  0.44],
       [-0.08,  0.52, -0.32],
       [ 0.28, -0.32,  0.12]])
```

เวกเตอร์และเมทริกซ์คณิตศาสตร์เชิงสัญลักษณ์

เอลิเมนต์ในเวกเตอร์และเมทริกซ์อาจอยู่ในรูปแบบของสัญลักษณ์ได้ ซอฟต์แวร์ไพธอนมีศักยภาพในการดำเนินการกับเวกเตอร์และเมทริกซ์เหล่านี้ได้ในทำนองเดียวกันกับเมื่อเอลิเมนต์นั้นอยู่ในรูปแบบของตัวเลข แพ็กเกจ SymPy บรรจุคำสั่งและฟังก์ชันเพื่อใช้จัดการกับสัญลักษณ์ เราต้อง Import แพ็กเกจนี้เข้ามาก่อนการใช้งานดังนี้

In [81]:
```
import sympy as sym
```

สมมติว่าเรามีเมทริกซ์ P และ Q ขนาด (2×2) ดังต่อไปนี้

$$[P] = \begin{bmatrix} 1 & 2x \\ 3x^2 & 4x^3 \end{bmatrix}$$

และ

$$[Q] = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

เราต้องระบุว่าตัวแปรใดเป็นสัญลักษณ์ก่อน แล้วจึงกำหนดเมทริกซ์ P และ Q ดังนี้

In [82]:
```
x = sym.symbols("x")
```

In [83]:
```
a, b, c, d = sym.symbols("a, b, c, d")
```

In [84]:
```
P = sym.Matrix([[1, 2*x], [3*x**2, 4*x**3]])
```

In [85]:
```
P
```

Out[85]:
$$\begin{bmatrix} 1 & 2x \\ 3x^2 & 4x^3 \end{bmatrix}$$

In [86]:
```
Q = sym.Matrix([[a, b], [c, d]])
```

In [87]:
```
Q
```

Out[87]:
$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

การบวกหรือลบกันระหว่างเมทริกซ์ เช่น

In [88]:
```
P + Q
```

Out[88]:
$$\begin{bmatrix} a+1 & b+2x \\ c+3x^2 & d+4x^3 \end{bmatrix}$$

In [90]:
```
3*Q - a*Q
```

Out[90]:
$$\begin{bmatrix} -a^2+3a & -ab+3b \\ -ac+3c & -ad+3d \end{bmatrix}$$

การคูณกันระหว่างเมทริกซ์ เช่น

In [47]:
```
Q * Q
```

Out[47]:
$$\begin{bmatrix} a^2+bc & ab+bd \\ ac+cd & bc+d^2 \end{bmatrix}$$

การหาค่าดีเทอร์มิแนนต์ของเมทริกซ์ Q

In [48]:
```
sym.det(Q)
```

Out[48]:
$$ad - bc$$

In [92]:
```
e, f, g, h, i = sym.symbols('e, f, g, h, i')
A = sym.Matrix([ [a, b, c], [d, e, f], [g, h, i] ])
A
```

Out[92]:
$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

In [93]:
```
sym.det(A)
```

Out[93]:
$$aei - afh - bdi + bfg + cdh - ceg$$

การหาค่าผลรวมของเอลิเมนต์ในแนวทแยงกลางของเมทริกซ์ Q

In [95]:
```
sym.trace(Q)
```

Out[95]:
$$a + d$$

การหาค่าอนุพันธ์อันดับที่หนึ่งของเมทริกซ์ P

In [98]:
```
sym.diff(P, x)
```

Out[98]:
$$\begin{bmatrix} 0 & 2 \\ 6x & 12x^2 \end{bmatrix}$$

การหาค่าอนุพันธ์อันดับที่สามของเมทริกซ์ P

In [51]:
```
sym.diff(P, x, 3)
```

Out[51]:
$$\begin{bmatrix} 0 & 0 \\ 0 & 24 \end{bmatrix}$$

การอินทิเกรตเมทริกซ์ P

In [101…
```
sym.integrate(P, x)
```

Out[101…
$$\begin{bmatrix} x & x^2 \\ x^3 & x^4 \end{bmatrix}$$

การอินทิเกรตเมทริกซ์ P จากลิมิต 0 ถึง 1

In [53]:
```
sym.integrate(P, (x, 0, 1))
```

Out[53]: $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$

อนึ่งเราสามารถเขียนคำสั่งในการแก้ปัญหาทางคณิตศาสตร์เป็น Python Scripts ก็จะช่วยให้เราแก้ปัญหาลักษณะเดียวกันได้อย่างอัตโนมัติ

บทสรุป

บทนี้นำเสนอการสร้างเวกเตอร์และเมทริกซ์เพื่อใช้งานในซอฟต์แวร์ไพธอน เวกเตอร์ และเมทริกซ์ เหล่านี้เป็นอาร์เรย์ซึ่งใช้สำหรับการเก็บค่าชุดตัวเลขและสัญลักษณ์ จากตัวอย่างที่นำเสนอจะเห็นได้ว่าการสร้างเวกเตอร์และเมทริกซ์นั้นสามารถทำได้โดยง่ายบนซอฟต์แวร์ไพธอน จากนั้นจึงนำเสนอวิธีการดำเนินการทางคณิตศาสตร์สำหรับเวกเตอร์และเมทริกซ์ที่ไม่ว่าจะเป็นการบวก การลบ การคูณ การทรานสโพส การหาค่าดีเทอร์มิแนนต์ และการหาเมทริกซ์ผกผัน เป็นต้น สุดท้ายเป็นการนำเสนอศักยภาพของซอฟต์แวร์ไพธอนที่ช่วยให้เราดำเนินการทางคณิตศาสตร์เชิงสัญลักษณ์ เช่น การหาค่าอนุพันธ์และการอินทิเกรตสำหรับเอลิเมนต์ภายในเวกเตอร์หรือเมทริกซ์ได้โดยสะดวก พื้นฐานความเข้าใจเหล่านี้เป็นสิ่งจำเป็นในการประดิษฐ์โปรแกรมคอมพิวเตอร์เพื่อใช้แก้ปัญหาในระดับสูงต่อไป

แบบฝึกหัด

กำหนดให้ $\vec{i}, \vec{j}$ และ $\vec{k}$ เป็นเวกเตอร์หน่วย (Unit Vector) ในปริภูมิเวกเตอร์สามมิติตามแนวแกน x, y และ z ตามลำดับ

1. กำหนดให้เวกเตอร์ $\vec{u} = 3\vec{i} + 4\vec{j}$ และ $\vec{v} = \vec{i} - 5\vec{j}$ เวกเตอร์ $\vec{u}$ และ $\vec{v}$ สามารถเขียนในอีกรูปแบบได้ดังนี้

$$\vec{u} = \begin{bmatrix} 3 \\ 4 \end{bmatrix} \quad \text{และ} \quad \vec{v} = \begin{bmatrix} 2 \\ -5 \end{bmatrix}$$

   จงใช้ไพธอนเพื่อคำนวณหา

   A. $\vec{u} + \vec{v}$
   B. $\vec{u} - \vec{v}$
   C. $\vec{u} * \vec{v}$
   D. $\log \vec{u}$
   E. $\vec{u} \cdot \vec{v}$
   F. $\vec{v} \cdot \vec{u}$
   G. $\vec{u} \times \vec{v}$
   H. $\vec{v} \times \vec{u}$

1. กำหนดให้เวกเตอร์ $\vec{u} = 4\vec{i} + 2\vec{j}, \vec{v} = -\vec{i} + 5\vec{j}$ และ $\vec{w} = -3\vec{i} + 2\vec{j}$ เวกเตอร์ $\vec{u}, \vec{v}$ และ $\vec{w}$ สามารถเขียนในอีกรูปแบบได้ดังนี้

$$\vec{u} = \begin{bmatrix} 4 \\ 2 \end{bmatrix} \ ; \ \vec{v} = \begin{bmatrix} -1 \\ 5 \end{bmatrix} \ ; \ \vec{w} = \begin{bmatrix} -3 \\ 2 \end{bmatrix}$$

จงใช้ไพธอนเพื่อคำนวณหา

A. $\vec{u} + \vec{v}$
B. $\vec{v} \cdot \vec{w}$
C. $\vec{v} \cdot \vec{v}$
D. $\vec{u} \cdot (3\vec{w})$
E. $\vec{u} \cdot (\vec{v} + \vec{w})$
F. $(-\vec{w}) \cdot \vec{u}$
G. $\vec{u} \times (\vec{v} + \vec{w})$
H. $(\vec{v} + \vec{w}) \times \vec{u}$

1. กำหนดให้เวกเตอร์ $\vec{u} = 2\vec{i} - 5\vec{j} + 7\vec{k}$ และ $\vec{v} = -\vec{i} + 4\vec{j} - 3\vec{k}$ เวกเตอร์ $\vec{u}$, และ $\vec{v}$ นี้สามารถเขียนในอีกรูปแบบได้ดังนี้

$$\vec{u} = \begin{bmatrix} 2 \\ -5 \\ 7 \end{bmatrix} \text{ และ } \vec{v} = \begin{bmatrix} -1 \\ 4 \\ -3 \end{bmatrix}$$

จงใช้ไพธอนเพื่อคำนวณหา

A. $\vec{u} + \vec{v}$
B. $\vec{u} * \vec{v}$
C. $\vec{u} \cdot \vec{v}$
D. $\vec{u} \times \vec{v}$
E. $(\vec{u} + \vec{v}) \cdot \vec{u}$
F. $(\vec{u} - \vec{v}) \times \vec{v}$
G. $\dfrac{\vec{v} - \vec{w}}{\sqrt{\vec{u} \cdot \vec{u}}}$ 1. $\left(\dfrac{\vec{u} \cdot \vec{v}}{\vec{v} \cdot \vec{v}}\right) * \vec{w}$

1. กำหนดให้เวกเตอร์ $\vec{u} = a\vec{i} + b\vec{j} + c\vec{k}$ และ $\vec{v} = d\vec{i} + e\vec{j} + j\vec{k}$ โดย $a, b, c, d, e, f \in \mathbb{R}$ เวกเตอร์ $\vec{u}$, และ $\vec{v}$ นี้สามารถเขียนในอีกรูปแบบได้ดังนี้

$$\vec{u} = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \quad \text{และ} \quad \vec{v} = \begin{bmatrix} d \\ e \\ f \end{bmatrix}$$

จงใช้ไพธอนเพื่อแสดงให้เห็นว่า

A. $\vec{u} \cdot \vec{v} = \vec{v} \cdot \vec{u}$
B. $\vec{u} \times \vec{v} = -\vec{v} \times \vec{u}$
C. $\vec{u} \times \vec{u} = \vec{v} \times \vec{v} = 0$
D. $\vec{u} \cdot (\vec{u} \times \vec{v}) = \vec{v} \cdot (\vec{u} \times \vec{v}) = 0$

1. กำหนดให้เวกเตอร์ $\vec{u}, \vec{v}$ และ $\vec{w}$ ในโคออร์ดิเนต 3 มิติ ซึ่งมีขนาดและทิศทางในรูปแบบดังนี้

$$\vec{u} = \begin{bmatrix} 7 \\ 4 \\ -1 \end{bmatrix}; \vec{v} = \begin{bmatrix} -5 \\ 3 \\ 8 \end{bmatrix}; \vec{w} = \begin{bmatrix} 9 \\ -6 \\ 2 \end{bmatrix}$$

จงใช้ไพธอนเพื่อพิสูจน์สมการต่อไปนี้

A. $\vec{u} \cdot \vec{v} = \vec{v} \cdot \vec{u}$
B. $\vec{u} \times \vec{v} = -\vec{v} \times \vec{u}$
C. $\vec{u} \cdot (\vec{v} + \vec{w}) = \vec{u} \cdot \vec{v} + \vec{u} \cdot \vec{w}$
D. $\vec{u} \times (\vec{v} + \vec{w}) = (\vec{u} \times \vec{v}) + (\vec{u} \times \vec{w})$
E. $(\vec{u} + \vec{v}) \times \vec{w} = (\vec{u} \times \vec{w}) + (\vec{v} \times \vec{w})$
F. $\vec{u} \times (\vec{v} + \vec{w}) = \vec{v} * (\vec{u} \cdot \vec{w}) - \vec{w} * (\vec{u} \cdot \vec{v})$
G. $(\vec{u} \times \vec{v}) \times \vec{w} = \vec{v} * (\vec{u} \cdot \vec{w}) - \vec{u} * (\vec{v} \cdot \vec{w})$

1. กำหนดเมทริกซ์

$$[A] = \begin{bmatrix} 8 & -3 \\ 11 & 4 \end{bmatrix} \;\; ; \;\; [B] = \begin{bmatrix} -2 & 7 \\ -5 & 9 \end{bmatrix}$$

จงใช้ไพธอนเพื่อคำนวณหา

A. $[A]^{-1}$ และ $[B]^T$
B. $|\,[A]\,|$ และ $|\,[B]\,|$
C. $[A] + [B]$ และ $[A]^T + [B]^{-1}$
D. $[A]\,[A]^{-1}$ และ $[A]\,[A]^T$
E. $[A] \cdot [B]$ และ $[A] * [B]$

1. กำหนดเมทริกซ์ในรูปแบบของสัญลักษณ์

$$[A] = \begin{bmatrix} p & q \\ q & p \end{bmatrix} \quad ; \quad [B] = \begin{bmatrix} r & s \\ s & r \end{bmatrix}$$

จงใช้ไพธอนเพื่อคำนวณหาผลลัพธ์ตามข้อย่อยในปัญหาข้อ 6

1. กำหนดเมทริกซ์

$$[A] = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \quad ; \quad [B] = \begin{bmatrix} 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{bmatrix}$$

จงใช้ไพธอนเพื่อคำนวณหาผลลัพธ์ตามข้อย่อยในปัญหาข้อ 6

1. กำหนดเมทริกซ์ในรูปแบบของสัญลักษณ์

$$[A] = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & a_9 \end{bmatrix} \quad ; \quad [B] = \begin{bmatrix} b_1 & b_2 & b_3 \\ b_4 & b_5 & b_6 \\ b_7 & b_8 & b_9 \end{bmatrix}$$

จงใช้ไพธอนเพื่อคำนวณหาผลลัพธ์ตามข้อย่อยในปัญหาข้อ 6 จากนั้นให้วิจารณ์ข้อดีและข้อเสียของการใช้สัญลักษณ์แทนตัวเลขดังเช่นที่ใช้ในข้อ 8

1. เมทริกซ์ฮิลเบิร์ด (Hilbert Matrix) คือเมทริกซ์จัตุรัสซึ่งค่าในตำแหน่งแถวนอนที่ i และแถวตั้งที่ j คือ $\dfrac{1}{(i+j-1)}$ ยกตัวอย่างเช่น เมทริกซ์ฮิลเบิร์ด $[H]$ ขนาด 3×3 คือ

$$\begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{bmatrix}$$

จงใช้ไพธอนเพื่อหาค่าดีเทอร์มิแนนต์และเมทริกซ์ผกผันเมื่อเมทริกซ์ฮิลเบิร์ดมีขนาด

A. 3×3
B. 5×5
C. 10×10

สังเกตและวิจารณ์ถึงผลลัพธ์ที่ได้เมื่อเมทริกซ์ฮิลเบิร์ดมีขนาดใหญ่ขึ้น

1. จงใช้ไพธอนเพื่อแก้ระบบสมการพีชคณิตเชิงเส้น

(ก)

$$\begin{bmatrix} 4 & 3 & 8 \\ 9 & 5 & 1 \\ 2 & 7 & 6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 26 \\ 38 \\ 26 \end{bmatrix}$$

(ข)

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

แล้วเปรียบเทียบความซับซ้อนของผลลัพธ์ที่เกิดขึ้นในข้อ (ก) และ (ข) จากนั้นให้วิจารณ์ข้อดีและข้อเสียของการใช้สัญลักษณ์เมื่อเปรียบเทียบกับตัวเลข

1. จงหาค่าดีเทอร์มิแนนต์เมทริกซ์ผกผันของเมทริกซ์ต่อไปนี้

(ก)

$$\begin{bmatrix} 1 & 1 \\ -2 & 4 \end{bmatrix}$$

(ข)

$$\begin{bmatrix} -149 & -50 & -154 \\ 537 & 180 & 546 \\ -27 & -9 & -25 \end{bmatrix}$$

1. จงหาค่าดีเทอร์มิแนนต์เมทริกซ์ผกผันของเมทริกซ์ในรูปแบบของสัญลักษณ์ต่อไปนี้

(ก)

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

(ข)

$$\begin{bmatrix} a & b & c \\ b & a & c \\ c & b & a \end{bmatrix}$$

1. กำหนดให้เมทริกซ์

$$[A] = \begin{bmatrix} 2x & x^2 \\ x^3 & 7x \end{bmatrix}; [B] = \begin{bmatrix} e^x & \sin x \\ \cos x & x \end{bmatrix}$$

   จงใช้ไพธอนเพื่อหาค่าอนุพันธ์ต่อไปนี้

   A. $\frac{d}{dx}[A]$
   B. $\frac{d}{dx}[B]$
   C. $\frac{d}{dx}([A]+[B])$
   D. $\frac{d}{dx}([A]\,[B])$

1. จงใช้เมทริกซ์ $[A]$ และ $[B]$ ที่กำหนดให้ในข้อ 14 เพื่อหาผลลัพธ์จากการอินทิเกรตต่อไปนี้

   A. $\int [A]\,dx$
   B. $\int_a^b [B]\,dx$
   C. $\int ([A]+[B])dx$
   D. $\int_a^b ([A]\,[B])dx$

```
In [ ]:
```