In [1]:

```
from pyIClab import (
    IonChromatograph, Eluent, PEEKTubing,
    Dummy, Column, SwitchingValve,
    SampleLoop, ContaminatedPhreeqcSuppressor,
    Detector, DSM_CompleteEquilibriums, DSM_CEConstrutor,
    IonExchanger,
    )

from typing import Callable
import numpy as np
import pyIClab.ions as ions_
from pyIClab.engines.models import (
    _total_mix,
    _total_mix_analyte,
    )
```

In [2]:

```
def local_post_distrubute(model, /, *,
    mix_n: int,
    ):

    for _ in range(mix_n):
        _total_mix(model)
        if model.analyte in ['Cl[-1]', 'SO4[-2]']:
            _total_mix_analyte(model)
```

In [3]:

```
class LocalConstructor(DSM_CEConstrutor):


    def set_post_distribute(self):

        return local_post_distrubute

    def set_post_distribute_params(self):

        N = self.set_N()
        length = self.host.length.to('cm').magnitude
        target_N = round(length / 0.04)
        mix_n = round(np.log2(2*N / target_N)) + 1

        return {'mix_n': mix_n}
```

In [4]:

```
seawater = {
    'Cl[-1]': 0.5442580719335613e3,
    'SO4[-2]': 0.028151884607340295e3,
    'Br[-1]': 0.0008395248249145663e3,
    'F[-1]': 6.82248096476202e-02,
    'OH[-1]': 8.207440112057654e-03,
    'NO3[-1]': 0.016,
    'NO2[-1]': 0.005,
    }
```

In [5]:

```
tb0, tb1, tb2, tb3, tb4 = [PEEKTubing(length='30cm') for i in range(5)]
```

In [6]:

```
sp = IonExchanger.load('home_made.dat', directory='db')
column = Column(f'Homemade', length='15 cm', ID='4.6 mm')
column.pack(sp)
concentrator = Column('Concentrator', length='5.0cm', ID='.46cm')
concentrator.pack(sp)
```

In [7]:

```
eluent = Eluent('KOH', profile={'OH-': ((7, 18), (16, 32))})
```

In [8]:

```
sixport = SwitchingValve.SixPort()
tenport = SwitchingValve.TenPort()
```

In [9]:

```
loop = SampleLoop('Loop', '25uL')
```

In [10]:

```
suppressor = ContaminatedPhreeqcSuppressor('Suppressor')
detector = Detector('Detector')
```

In [11]:

```
eluent.assemble(tb0)
sixport.assemble(2, tb0)
sixport.assemble([1, 4], concentrator)
sixport.assemble(3, tb1)
tenport.assemble(6, tb1)
tenport.assemble(7, column) ###
column.assemble(suppressor)
suppressor.assemble(detector)
tenport.assemble(1, detector)
tenport.assemble(0, tb3)
sixport.assemble(0, tb3)
sixport.assemble(5, tb4)
tenport.assemble(3, tb4)
tenport.assemble([5, 8], loop)
sixport.switch('INJECT')
```

In [12]:

```
ic = IonChromatograph(
    name='Cycled-Column-Swicthing-IC',
    competing_ions=('OH-',),
    lockon=(sixport, tenport),
    reset_valves=False)
```

In [13]:

```
ic.namespace
```

Out[13]:

	type_identifier	name	module_instance
0	column	Concentrator	<Column "Concentrator" (4.6 × 50 mm)>
1	column	Homemade	<Column "Homemade" (4.6 × 150 mm)>
2	detector	Detector	<Detector "Detector">
3	eluent	KOH	<Eluent "KOH" Gradient(OH[-1]: 18.0 ~ 32.0 mM,...
4	loop	Loop	<Loop "Loop" 25 µL>
5	suppressor	Suppressor	<Suppressor "Suppressor">
6	valve	SixPort	(<Valve "SixPort"[0]>, <Valve "SixPort"[1]>, <...
7	valve	TenPort	(<Valve "TenPort"[0]>, <Valve "TenPort"[1]>, <...

In [14]:

```
ic.inject(seawater, module='loop')

ic.injection_table
```

Out[14]:

	accessory	Cl[-1]	SO4[-2]	Br[-1]	F[-1]	OH[-1]	NO3[-1]	NO2[-1]	K[+1]
0	<Loop "Loop" 25 µL>	544.258072	28.151885	0.839525	0.068225	0.008207	0.016	0.005	601.498798

In [15]:

```
ic.set_ModelConstructor(LocalConstructor, concentrator)
ic.set_ModelConstructor(LocalConstructor, column)
```

In [16]:

```
commands = '''
    0.0 min, tenport, inject
    0.0 min, sixport, inject
    0.5 min, tenport, load
    7.2 min, sixport, load
    11.8 min, sixport, inject
    18.7 min, sixport, load
    22.5 min, sixport, inject
    28.7 min, sixport, load
    33.0 min, sixport, inject
    '''

ic.reset_commands(commands)
df = ic.schedule
df
```

Out[16]:

	time	type_identifier	name	module_instance	action
0	0.0	valve	SixPort	(<Valve "SixPort"[0]>, <Valve "SixPort"[1]>, <...	INJECT
1	0.0	valve	TenPort	(<Valve "TenPort"[0]>, <Valve "TenPort"[1]>, <...	INJECT
2	0.5	valve	TenPort	(<Valve "TenPort"[0]>, <Valve "TenPort"[1]>, <...	LOAD
3	7.2	valve	SixPort	(<Valve "SixPort"[0]>, <Valve "SixPort"[1]>, <...	LOAD
4	11.8	valve	SixPort	(<Valve "SixPort"[0]>, <Valve "SixPort"[1]>, <...	INJECT
5	18.7	valve	SixPort	(<Valve "SixPort"[0]>, <Valve "SixPort"[1]>, <...	LOAD
6	22.5	valve	SixPort	(<Valve "SixPort"[0]>, <Valve "SixPort"[1]>, <...	INJECT
7	28.7	valve	SixPort	(<Valve "SixPort"[0]>, <Valve "SixPort"[1]>, <...	LOAD
8	33.0	valve	SixPort	(<Valve "SixPort"[0]>, <Valve "SixPort"[1]>, <...	INJECT

In [17]:

```
%%time
ic.start(tmax='45 min')
```

```
11:27:30 Activating <IC System "Cycled-Column-Swicthing-IC">...
    11:27:30 Configurating model paratemers...
    11:27:40 Building models...
    11:28:00 Injecting Samples...
        0.0 min: Execute Command -- <Valve "SixPort"> INJECT
        0.0 min: Execute Command -- <Valve "TenPort"> INJECT

        0.5 min: Execute Command -- <Valve "TenPort"> LOAD

        7.2 min: Execute Command -- <Valve "SixPort"> LOAD

        11.8 min: Execute Command -- <Valve "SixPort"> INJECT

        18.7 min: Execute Command -- <Valve "SixPort"> LOAD

        22.5 min: Execute Command -- <Valve "SixPort"> INJECT

        28.7 min: Execute Command -- <Valve "SixPort"> LOAD

        33.0 min: Execute Command -- <Valve "SixPort"> INJECT

11:44:32 IC simulation finished...

CPU times: user 16min 57s, sys: 6.14 s, total: 17min 3s
Wall time: 17min 2s
```

In [18]:

```
df = detector.get_signals(signal_type='conductivity')
x, y = df['time'], df['signal']
```

```
/Users/kennyzhang/miniconda3/envs/pyiclab/lib/python3.11/site-packages/pyIClab/assemblies/signals.py:255: UserW
arning: Compromised accuracy in the conductivity profiles for the following analytes: NO2[-1].
  warnings.warn(
```

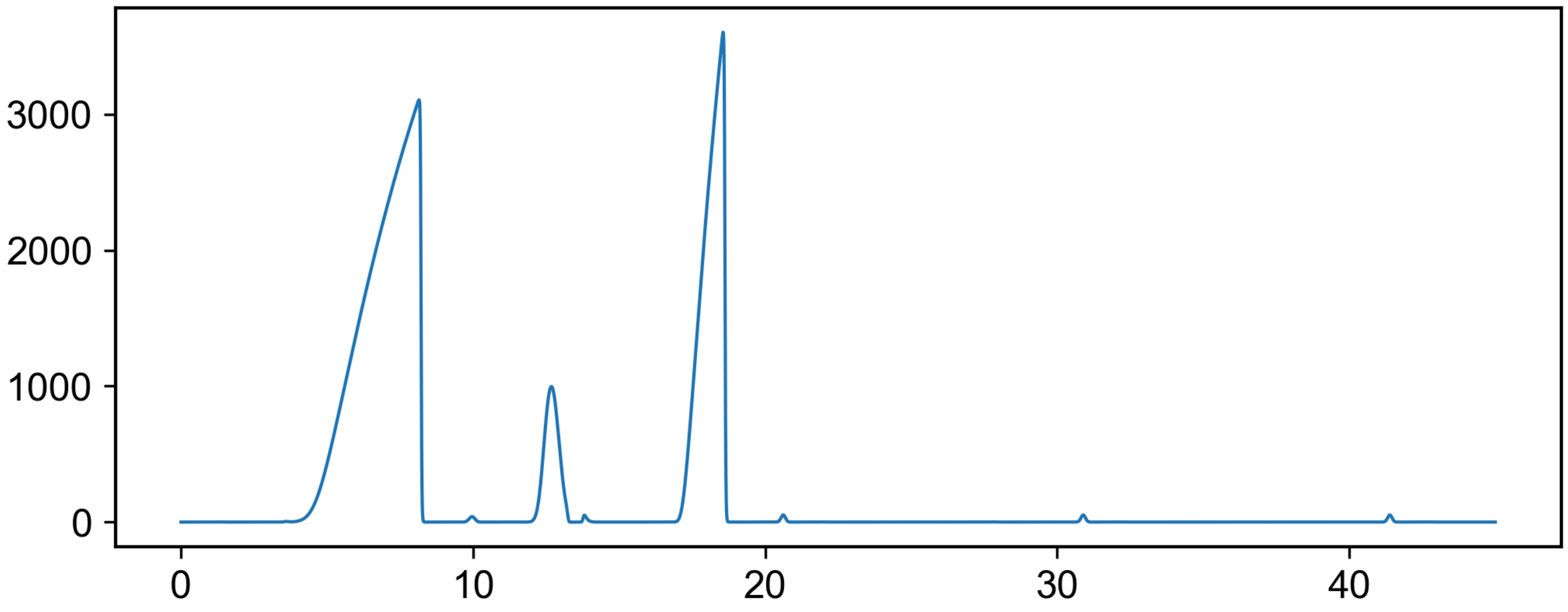
In [19]:

```
import matplotlib.pyplot as plt
from pyIClab.beadedbag import mpl_custom_rcconfig

plt.rcParams.update(mpl_custom_rcconfig)
fig, ax = plt.subplots()
ax.plot(x, y)
# ax.set(xlim=(30, 50))
# ax.set(ylim=(-.2, 1))
```

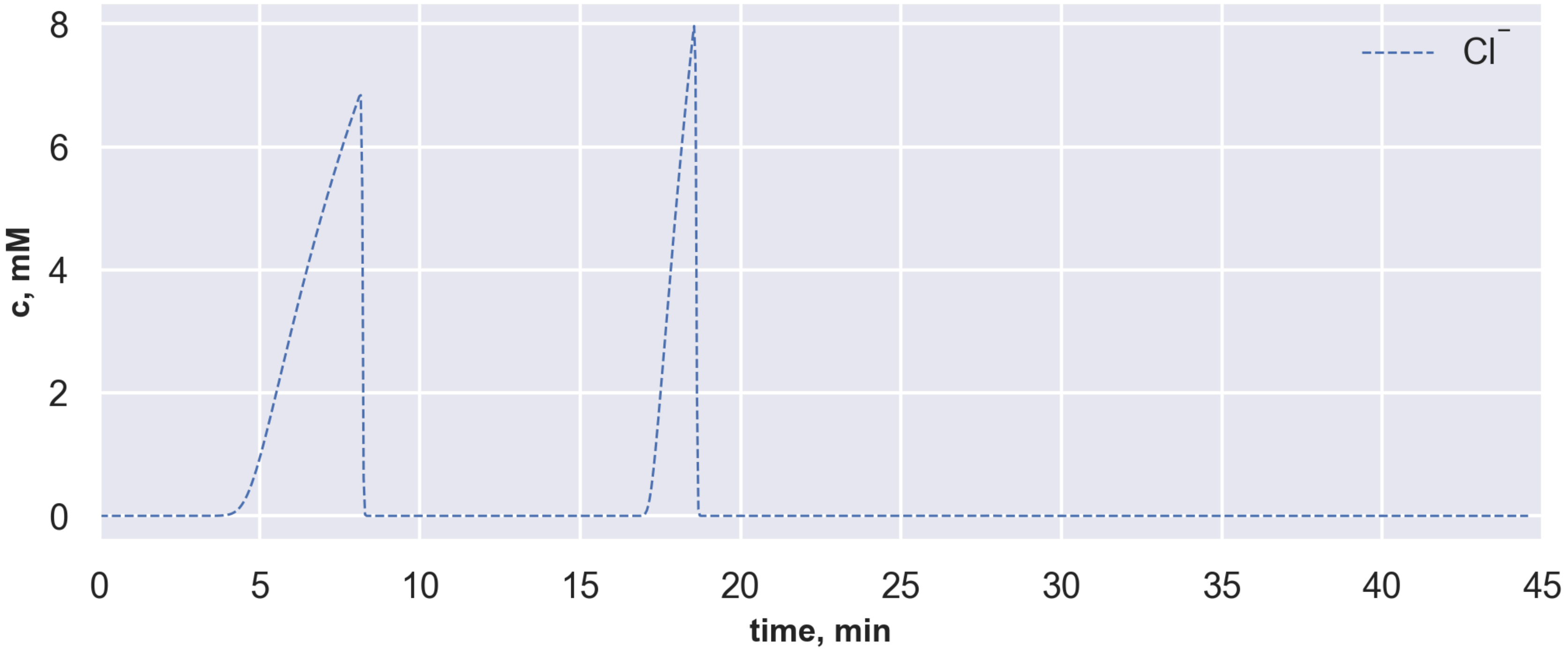
Out[19]:

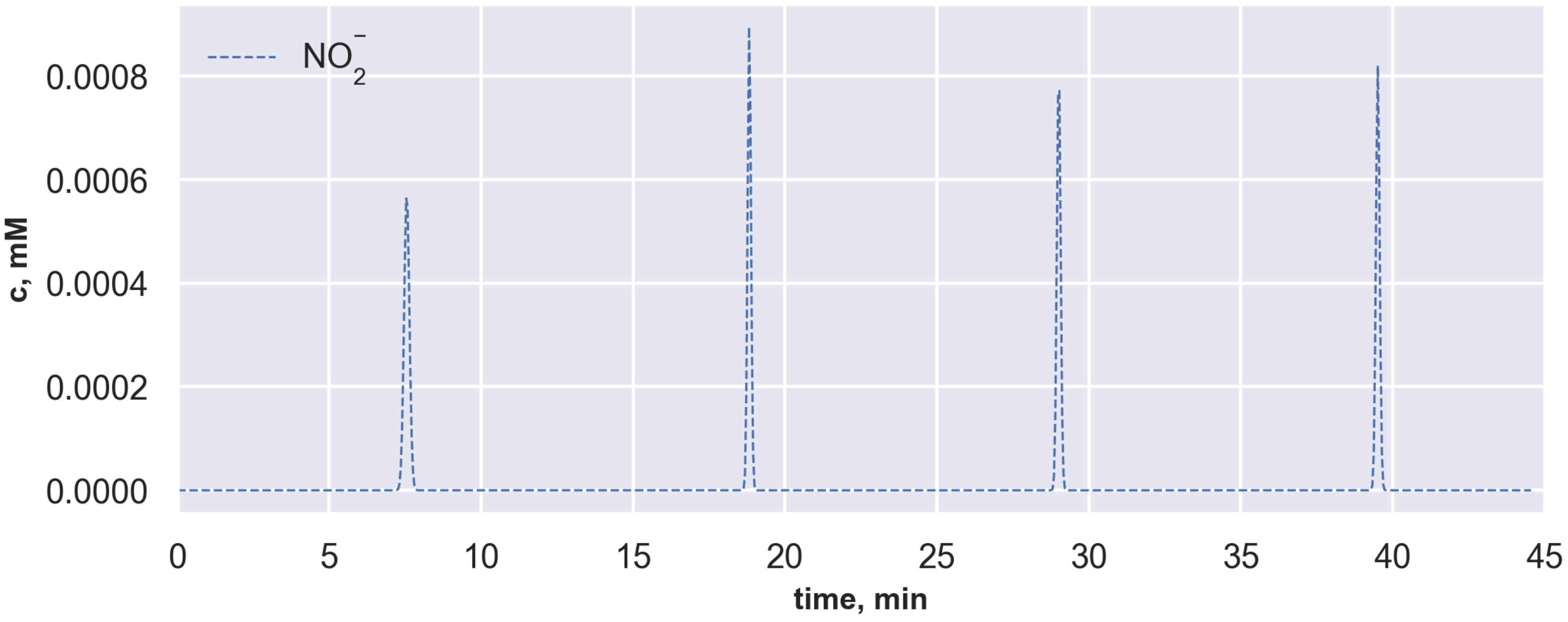
```
[<matplotlib.lines.Line2D at 0x1966d4c90>]
```

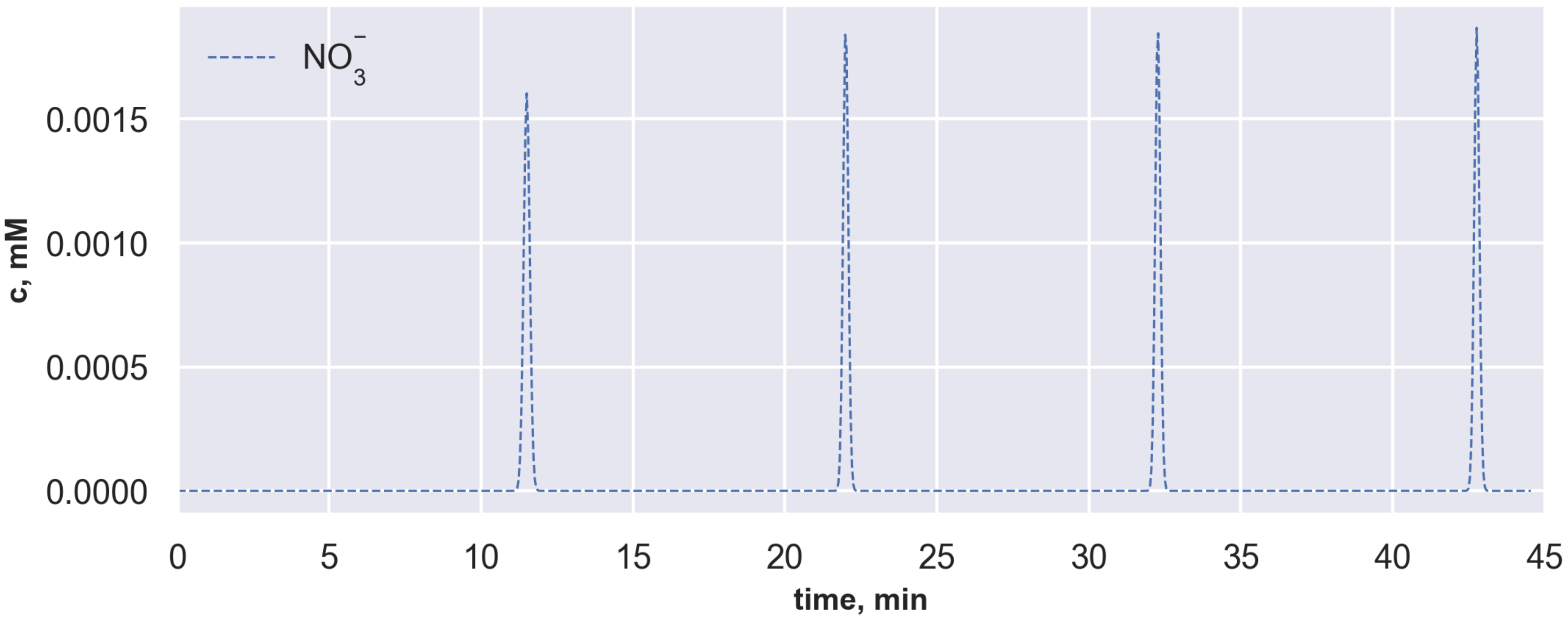


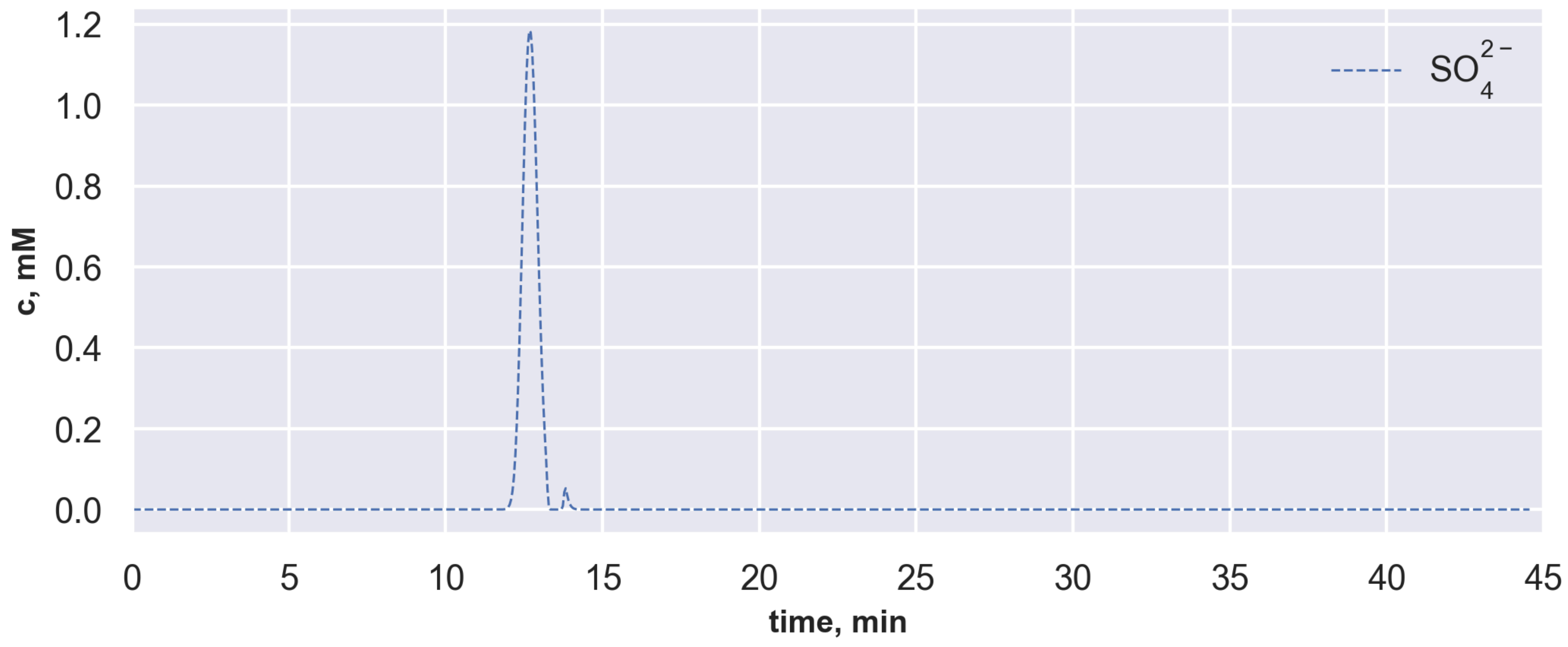
```
In [20]: detector.plot('Cl-')
         detector.plot('NO2-')
         detector.plot('NO3-')
         detector.plot('SO4-2')
```

```
Out[20]: (<Figure size 2400x900 with 1 Axes>,
          <Axes: xlabel='time, min', ylabel='c, mM'>)
```









In [21]:

```
detector(11.8)
```

Out[21]:

```
{'Br[-1]': array(0.),
 'Cl[-1]': array(9.78565502e-12),
 'F[-1]': array(0.),
 'NO2[-1]': array(0.),
 'NO3[-1]': array(2.10244731e-05),
 'SO4[-2]': array(0.00012755),
 'OH[-1]': array(1.64017097e-05),
 'H[+1]': array(0.00089771),
 'HCO3[-1]': array(0.00065193),
 'CO3[-2]': array(3.76102238e-08),
 'CO2': array(0.00124265)}
```

In [22]:

```
import pandas as pd
df = pd.DataFrame(data=dict(time=x, signal=y))
# df.to_csv('cycled column swicthing-18-30-32-homemade-A-int-charge.csv', index=False)
```

In [ ]: