Martin Iniguez de Onzono Muruaga Topic 06 CP

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

Exercise 1. Potassium channel

a) Write Python functions.

In [2]:
```
def alpha_n(V):
    return (0.01*(V+55)/(1-np.exp(-0.1*(V+55)))) # V in mV

def beta_n(V):
    return 0.125*np.exp(-0.0125*(V+65)) # V in mV

def ik(V,ek,n,gk):
    return gk*(n**4)*(V-ek) # mA

def dn(n,t,V):
    return alpha_n(V)*(1-n)-beta_n(V)*n
```

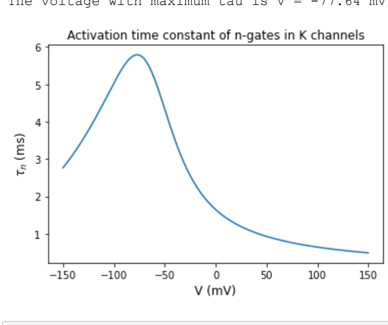
In [3]:
```
# Constants needed for the exercise

ek = -77 # mV
gk = 36e-9 # S
```

b) Plot the activation time constant and the steady-state activation.

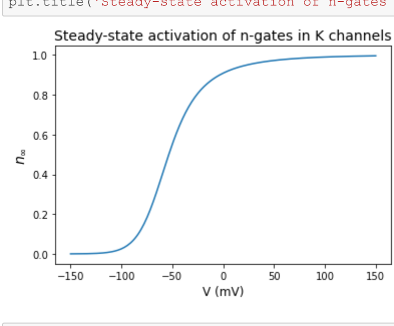
In [4]:
```
def tn(V):
    return 1/(alpha_n(V)+beta_n(V)) # ms

def n_inf(V):
    return tn(V)*alpha_n(V)

def euler(f_func, x_0, t_max, dt,V):
    iterat = int(np.floor(t_max/dt))
    x = np.zeros(iterat)
    x[0]=x_0

    for i in range(1,iterat):
        x[i] = x[i-1]+f_func(x[i-1],dt*(i-1),V[i-1]) *dt
    return x
```

In [5]:
```
V = np.linspace(-150,150,200)
tau_n = tn(V)
V_maxt = round(V[np.argmax(tau_n)],2)

plt.plot(V,tau_n)
plt.xlabel('V (mV)',fontsize=12);
plt.ylabel(r'$\tau_{n}$ (ms)',fontsize=12);
plt.title('Activation time constant of n-gates in K channels',fontsize=12);
print('The voltage with maximum tau is V = ' +str(V_maxt) + ' mV')
```

```
The voltage with maximum tau is V = -77.64 mV
```

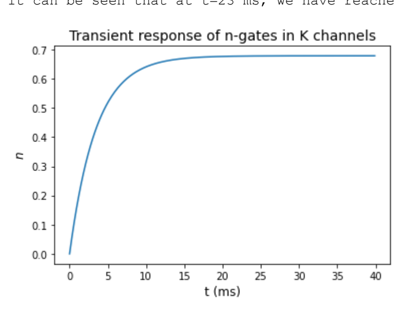


In [6]:
```
ninf = n_inf(V)

plt.plot(V,ninf)
plt.xlabel('V (mV)',fontsize=12);
plt.ylabel(r'$n_{∞}$',fontsize=14);
plt.title('Steady-state activation of n-gates in K channels',fontsize=14);
```



In [7]:
```
n0 = 0
dt = 0.1

t=np.arange(0,40,dt) # ms
n=np.zeros(len(t))
V_set = -40*np.ones(len(t)) #constant V (mV)

# Calculating n values using Euler's method
n_eu_b = euler(dn,n0,40,dt,V_set)

ind_b=np.array(np.where(t==23))
steady_n = round(n_eu_b[ind_b[0][0]],3)
print('It can be seen that at t=23 ms, we have reached the steady state, n∞=' + str(steady_n))

plt.plot(t,n_eu_b)
plt.xlabel('t (ms)',fontsize=12);
plt.ylabel(r'$n$',fontsize=12);
plt.title('Transient response of n-gates in K channels',fontsize=14);

## Do we need to do more than this(?)
```

```
It can be seen that at t=23 ms, we have reached the steady state, n∞=0.678
```

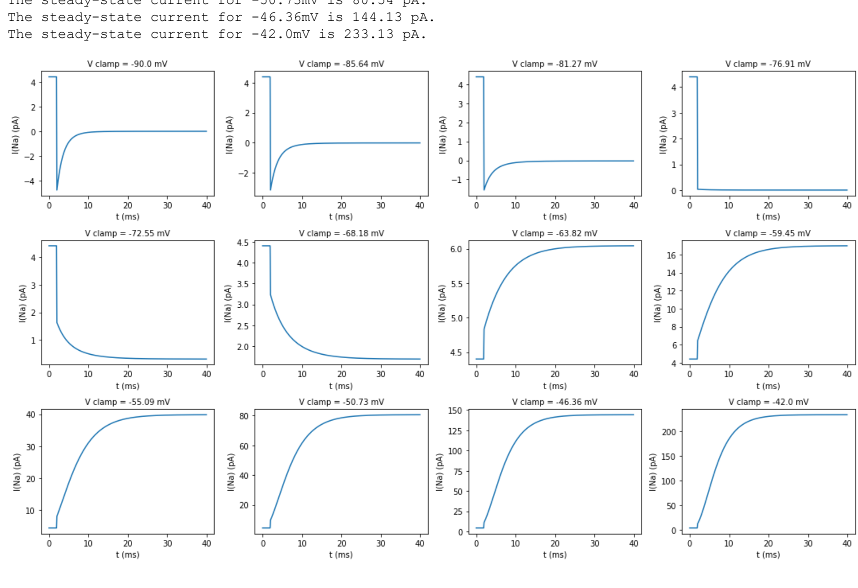


c) Voltage clamp. Simulate the current responses Ik to voltage steps under voltage-clamp conditions.

In [8]:
```
def V_clamp(Vc,t):
    ind=np.array(np.where(t==2)) # ms
    V_start = -65*np.ones((ind[0][0]))
    V_end = Vc*np.ones(len(t)-ind[0][0])
    V = np.concatenate((V_start,V_end))
    return V
```

In [9]:
```
n_0 = 0.3177
dt_c = 0.1
t=np.arange(0,40,dt_c) # ms
dim=12

Vc = np.linspace(-90,-42,dim)
    # I use as upper-bound -42 mV instead of 40, and a lower bound of -90 mV instead of -100 because we
 run into discontinuites in 2c if not.
I_k_1c = np.zeros((len(t),len(Vc)))

fig = plt.figure(figsize=(15,9))

for i in range(0,len(Vc)):

    V_c = V_clamp(Vc[i],t)

    # Calculating n values using Euler's method
    n_eu_1c = euler(dn,n_0,40,dt_c,V_c)

    I_k_1c[:,i] = ik(V_c,ek,n_eu_1c,gk)
    print('The steady-state current for ' + str(round(V_c[-1],2)) + 'mV is ' +str(round(I_k_1c[-1,i]*10
**9,2))+' pA.')
    fig.add_subplot(3, 4, i+1)

    plt.plot(t,I_k_1c[:,i]*10**9)# The current units before were mA, and now we set it to pA
    plt.xlabel('t (ms)',fontsize=10);
    plt.ylabel('I(Na) (pA)',fontsize=10);
    plt.title('V clamp = ' + str(round(Vc[i],2)) + ' mV',fontsize=10);

fig.tight_layout()
```

```
The steady-state current for -90.0mV is -0.01 pA.
The steady-state current for -85.64mV is -0.02 pA.
The steady-state current for -81.27mV is -0.03 pA.
The steady-state current for -76.91mV is 0.0 pA.
The steady-state current for -72.55mV is 0.32 pA.
The steady-state current for -68.18mV is 1.69 pA.
The steady-state current for -63.82mV is 6.04 pA.
The steady-state current for -59.45mV is 16.98 pA.
The steady-state current for -55.09mV is 39.82 pA.
The steady-state current for -50.73mV is 80.54 pA.
The steady-state current for -46.36mV is 144.13 pA.
The steady-state current for -42.0mV is 233.13 pA.
```



What can be learned from this experiment? What is the predicted effect of the potassium current on the membrane potential? Explain the obtained results by referring to the plots of n_inf and tau_n.

It can be seen that, when applying clamp voltages smaller than Ek, the current induced by the potassium ions in steady-state is zero after a small negative spike. However, when applying a Voltage clamp higher than that, the potassium ions try to compensate and start leaving the neuron.

From Ek to -65 mV, the current holds a exponential decrease in time (although the steady-state is a positive current).

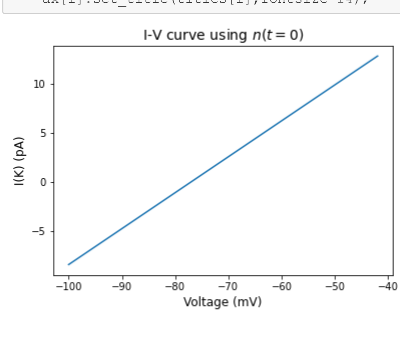
After -65 mV, the current increases (in a negative exponential decay manner) from its starting value.

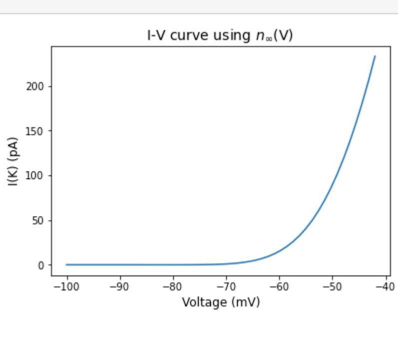
That can be seen in the tau plot in 1b, in which we can see that the tau values start decreasing after the tau(V=Ek).

Therefore, the current from inwards to outwards (direction set by convention) is positive, as positive ions flow outwards the neuron.

d) Current-voltage curves.

In [10]:
```
length_d = 50

V_d = np.linspace(-100,-42,length_d) # Until -42 because discontinuities appear if not at 2d

# Preparing two different sets of n's = one column of n at t=0, and one of the different n(V) at their
 different steady-state
n=np.full(length_d,n_0)
n = np.vstack((n,n_inf(V_d))).T

I_k_1d = np.zeros((length_d,n.shape[1]))

fig, ax = plt.subplots(figsize=(11, 4),nrows=1, ncols=2, constrained_layout=True)
titles=['I-V curve using $n(t=0)$',r'I-V curve using $n_{∞}$(V) ']

for i in range(0,n.shape[1]):

    I_k_1d[:,i] = ik(V_d,ek,n[:,i],gk) # The current units before were mA, and now we set it to pA, so
 it can be understood

    ax[i].plot(V_d,I_k_1d[:,i]*10**9)
    ax[i].set_xlabel('Voltage (mV)',fontsize=12);
    ax[i].set_ylabel('I(K) (pA)',fontsize=12);
    ax[i].set_title(titles[i],fontsize=14);
```





Should we use a specific n for the second I-V curve or calculate each n with respect to voltage????????

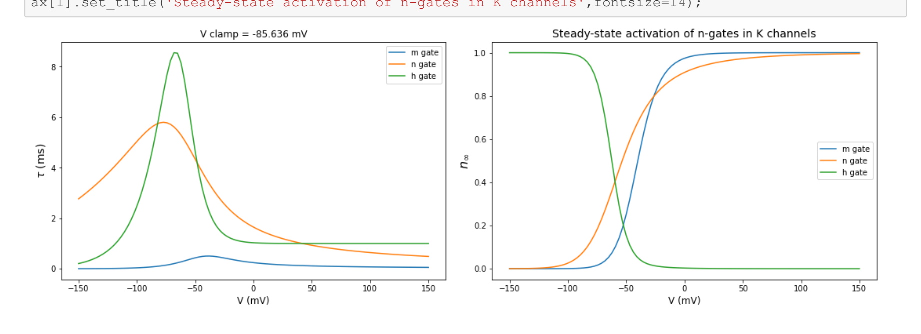
Exercise 2. Sodium ion channel.

In [11]:
```
def Ina(V,Ena,m,h,gna):
    return gna*(m**3)*h*(V-Ena)

def alpha_m(V):
    return (0.1*(V+40)/(1-np.exp(-0.1*(V+40)))) # mV

def beta_m(V):
    return 4*np.exp(-0.0556*(V+65)) # mV

def alpha_h(V):
    return 0.07*np.exp(-0.05*(V+65)) # mV

def beta_h(V):
    return 1/(1+np.exp(-0.1*(V+35))) # mV
```

a) Write all functions and plot the steady-state activation and time constants.

In [12]:
```
def tm(V):
    return 1/(alpha_m(V)+beta_m(V))

def th(V):
    return 1/(alpha_h(V)+beta_h(V))

def m_inf(V):
    return tm(V)*alpha_m(V)

def h_inf(V):
    return th(V)*alpha_h(V)
```

```
In [13]: V = np.linspace(-150,150,100)

         # Computing the different times of activation of each gate
         tau_m = tm(V)
         tau_n = tn(V)
         tau_h = th(V)

         fig, ax = plt.subplots(figsize=(15, 5),nrows=1, ncols=2, constrained_layout=True)

         ax[0].plot(V,tau_m,label='m gate')
         ax[0].plot(V,tau_n,label='n gate')
         ax[0].plot(V,tau_h,label='h gate')
         ax[0].set_xlabel('V (mV)',fontsize=12);
         ax[0].set_ylabel(r'$\tau$ (ms)',fontsize=14);
         ax[0].set_title('V clamp = ' + str(round(Vc[i],3)) + ' mV',fontsize=12);
         ax[0].legend();


         # Computing the different stea
```
data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAABEAAAAFwCAYAAAC4vQ5FAAAABHNCSVQICAgIfAhkiAAAAAlwSFlzAAALEgAACxIB0t1+/AAAADh0RVh0U29mdHdhcmUAbWF0cGxvdGxpYiB2ZXJzaW9uMy4yLjIsIGh0dHA6Ly9tYXRwbG90bGliLm9yZy+WH4yJAAAgAElEQVR4nOzdeXxU9dn//9c1k5WsJIAgWwKyhB0MCIpbURG1emtdqtC6VG1dlvlV/t95V22prrbV1qd53i1Zba62o1aK1dUWslVpRNlEx7BD2JSQkZCH75/fHmYQhJCGBJDOTvJ8PzmPmnPOZc65ZZsicuc5nMeccIiIiIiIiIiKd
...
IiHR6EdEJao8ePVxGRkaowxAREZFGLF26dI9zrmeo4+gIuiYREREJb81dl0REAiQjI4MlS5aEOwRERFphJltCnUMHUXXJCIiIuGtue
...
IiIiIpLwlAARERERERERkYSnBIiIiIiIiIiIJDwlQERERERERERQk4SkBIiIiIiIiIiIJ7/8DMtcD/+za/2sAAAAASUVORK5CYII=
```
states of each gate
         ninf = n_inf(V)
         minf = m_inf(V)
         hinf = h_inf(V)

         ax[1].plot(V,minf,label='m gate')
         ax[1].plot(V,ninf,label='n gate')
         ax[1].plot(V,hinf,label='h gate')
         ax[1].legend();
         ax[1].set_xlabel('V (mV)',fontsize=12);
         ax[1].set_ylabel(r'$n_{\infty}$',fontsize=16);
         ax[1].set_title('Steady-state activation of n-gates in K channels',fontsize=14);
```



Compare the results to the dynamics of the n variable. Which variables are activated/deactivated by depolarisation? Which variables are fast/slow?

Regarding tau: We can see that the tau_m is the fastest, as it has a very fast activation time for all voltages. The n gate and the h gate are similarly slow. However, the tau_n have a more broad distribution, while the tau_h is constant over V apart from the peak in [-100,-40] mV.

Regarding n_inf: The m gate has a very similar distribution than the one from the n gates (they remain active in higher voltages, i.e depolarization). Nevertheless, the m-gate has a delayed but faster transition from 0 to 1. The h gate, on the other hand, is activated on low voltages and deactivated at highest ones (deactivated by depolarisation).

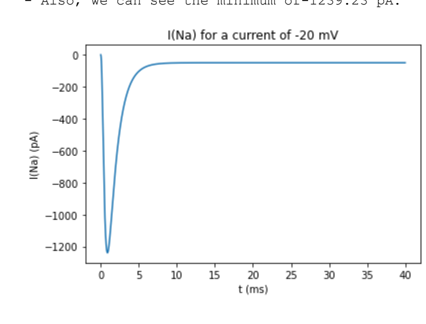
b) Simulate the sodium current

```
In [14]: def dm(m,t,V):
             return alpha_m(V)*(1-m)-beta_m(V)*m

         def dh(h,t,V):
             return alpha_h(V)*(1-h)-beta_h(V)*h
```

```
In [15]: Ena = 50 # mV
         gna = 120e-9 # S
```

```
In [16]: m0 = 0.0529
         h0 = 0.5961
         dt = 0.001

         t=np.arange(0,40,dt) # ms
         n=np.zeros(len(t))
         V_set = -20*np.ones(len(t)) #constant V (mV)

         # Calculating m and h values using Euler's method

         m_eu_b = euler(dm,m0,40,dt,V_set)
         h_eu_b = euler(dh,h0,40,dt,V_set)

         I_na_b = Ina(V_set,Ena,m_eu_b,h_eu_b,gna)

         ind_2b_1=np.array(np.where(t==15))
         ind_2b_2=np.argmin(I_na_b)

         steady_na = I_na_b[ind_2b_1[0][0]]
         min_na = I_na_b[ind_2b_2]

         print('It can be seen that:\n- At t=15 ms, we have reached the sodium current steady state, Ina = '+
               str(round(steady_na*10**9,3)) + ' pA.\n- Also, we can see the minimum of'+str(round(min_na*10**9,
         3)) +' pA.')

         plt.plot(t,I_na_b*10**9);
         plt.xlabel('t (ms)',fontsize=10);
         plt.ylabel('I(Na) (pA)',fontsize=10);
         plt.title('I(Na) for a current of -20 mV',fontsize=12);
```

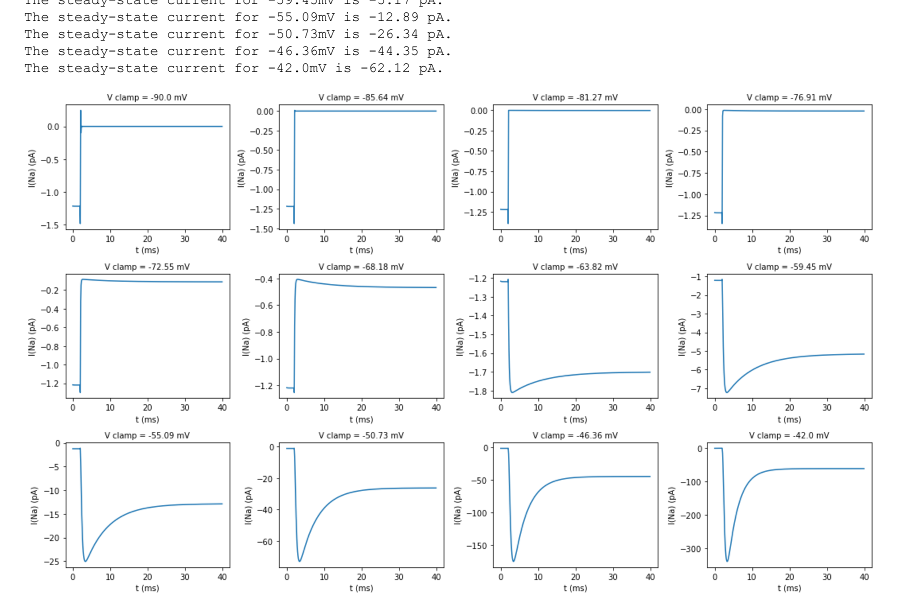
```
It can be seen that:
- At t=15 ms, we have reached the sodium current steady state, Ina = -50.499 pA.
- Also, we can see the minimum of-1239.23 pA.
```



c) Simulate the voltage clamp experiment.

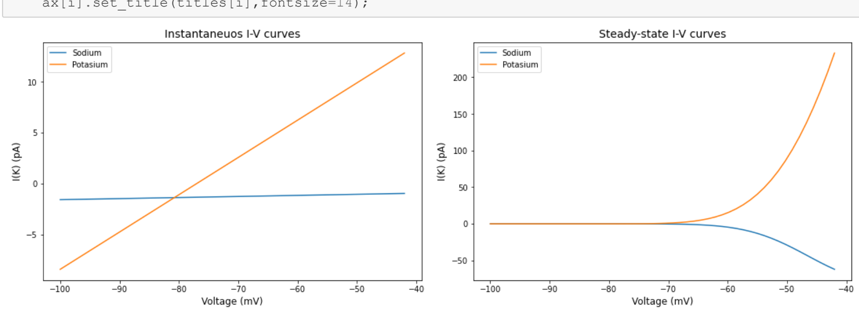
```
In [17]: t=np.arange(0,40,dt_c) # ms
         dim =12

         Vc = np.linspace(-90,-42,dim) # 90 and 42 mV instead of 100 and40 because we run into a discontinuity
         I_na_2c = np.zeros((len(t),len(Vc)))

         fig = plt.figure(figsize=(15,9))

         for i in range(0,len(Vc)):

             V_2c = V_clamp(Vc[i],t)

             m_eu_2c = euler(dm,m0,40,dt_c,V_2c)
             h_eu_2c = euler(dh,h0,40,dt_c,V_2c)

             I_na_2c[:,i] = Ina(V_2c,Ena,m_eu_2c,h_eu_2c,gna)
             print('The steady-state current for ' + str(round(V_2c[-1],2)) + 'mV is ' +str(round(I_na_2c[-1,i]*
         10**9,2))+' pA.')

             fig.add_subplot(3, 4, i+1)

             plt.plot(t,I_na_2c[:,i]*10**9)# The current units before were mA, and now we set it to pA
             plt.xlabel('t (ms)',fontsize=10);
             plt.ylabel('I(Na) (pA)',fontsize=10);
             plt.title('V clamp = ' + str(round(Vc[i],2)) + ' mV',fontsize=10);

         fig.tight_layout()
```

```
The steady-state current for -90.0mV is -0.0 pA.
The steady-state current for -85.64mV is -0.0 pA.
The steady-state current for -81.27mV is -0.0 pA.
The steady-state current for -76.91mV is -0.02 pA.
The steady-state current for -72.55mV is -0.11 pA.
The steady-state current for -68.18mV is -0.47 pA.
The steady-state current for -63.82mV is -1.7 pA.
The steady-state current for -59.45mV is -5.17 pA.
The steady-state current for -55.09mV is -12.89 pA.
The steady-state current for -50.73mV is -26.34 pA.
```



d) Plot the instantaneous and steady-state I-V curves for sodium channels.

In [18]:
```python
length_d = 50

V_2d = np.linspace(-100,-42,length_d) # Discontinuities at -40 mV, so we use as upper bound -40 mV

m = np.full(length_d,m0)
m = np.vstack((m,m_inf(V_2d))).T

h = np.full(length_d,h0)
h = np.vstack((h,h_inf(V_2d))).T

I_na_2d = np.zeros((length_d,m.shape[1]))

fig, ax = plt.subplots(figsize=(15, 5),nrows=1, ncols=2, constrained_layout=True)
titles=['Instantaneuos I-V curves','Steady-state I-V curves']

for i in range(0,m.shape[1]):
    I_na_2d[:,i] = Ina(V_2d,Ena,m[:,i],h[:,i],gna)

    ax[i].plot(V_d,I_na_2d[:,i]*10**9,label='Sodium') # The current units before were mA, and now we se
t it to pA
    ax[i].plot(V_d,I_k_1d[:,i]*10**9,label='Potasium') # The current units before were mA, and now we s
et it to pA

    ax[i].legend()
    ax[i].set_xlabel('Voltage (mV)',fontsize=12);
    ax[i].set_ylabel('I(K) (pA)',fontsize=12);
    ax[i].set_title(titles[i],fontsize=14);
```



Compare with 1d) and discuss the results.

For instantaneous I-V curves:

It can be seen that the current is always negative for Na channels, and the I-V curve has a very small slope compared to the one of the K channels, which follow a steep slope distribution ranging from negative to positive values.

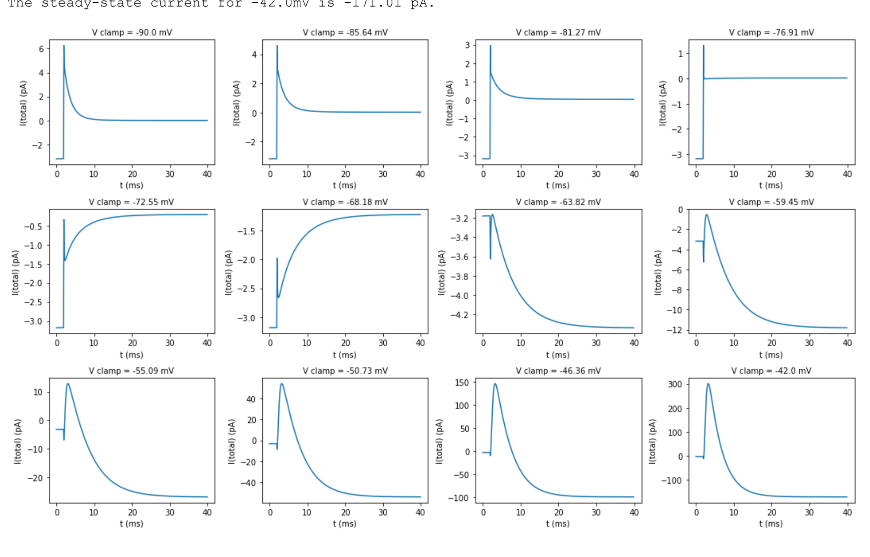
For steady-state I-V curves:

Both currents are null for both channels in case of low voltages. However, when approx. -70 mV are met, exponential curves arise, being the one from the K channels steeper and positive (outflow of K), and the one from Na channels negative (inflow of Na).

e) What happens if you repeat the voltage clamp experiments from before (initial conditions as before) but this time investigating the (negative) sum of both currents (I)? Plot the time course for varying command voltages (similar ranges as for the sodium experiment). Please speculate how this relates to neural activity.

In [19]:
```python
I_sum=-(I_k_1c+I_na_2c)

fig = plt.figure(figsize=(15,9))

for i in range(0,len(Vc)):

    V_2c = V_clamp(Vc[i],t)

    print('The steady-state current for ' + str(round(V_2c[-1],2)) + 'mV is ' +str(round(I_sum[-1,i]*10
**9,2))+' pA.')

    fig.add_subplot(3, 4, i+1)

    plt.plot(t,I_sum[:,i]*10**9)# The current units before were mA, and now we set it to pA
    plt.xlabel('t (ms)',fontsize=10);
    plt.ylabel('I(total) (pA)',fontsize=10);
    plt.title('V clamp = ' + str(round(Vc[i],2)) + ' mV',fontsize=10);

fig.tight_layout()
```

```
The steady-state current for -90.0mV is 0.01 pA.
The steady-state current for -85.64mV is 0.02 pA.
The steady-state current for -81.27mV is 0.03 pA.
The steady-state current for -76.91mV is 0.02 pA.
The steady-state current for -72.55mV is -0.21 pA.
The steady-state current for -68.18mV is -1.23 pA.
The steady-state current for -63.82mV is -4.34 pA.
The steady-state current for -59.45mV is -11.81 pA.
The steady-state current for -55.09mV is -26.92 pA.
The steady-state current for -50.73mV is -54.2 pA.
The steady-state current for -46.36mV is -99.77 pA.
The steady-state current for -42.0mV is -171.01 pA.
```



Please speculate how this relates to neural activity.

It can be seen that for low clamping voltages (below -65 mV), the current after the first positive, instantaneous spike, goes to zero.

However, that changes from -65 mV on (as seen in the steady-state IV curves), in which we can see a positive spike (due to the m gates) followed by the exponential decay due to the potassium and h gates, which repolarize the membrane.