Laboratorium 1: Symulacje deterministyczne

Imię i nazwisko: Anna Bernard, Mikołaj Mieszko Charchuta **Grupa:** Symulanci **Data wykonania:** 18.05.2025

---

1. Wstęp

W tym sprawozdaniu przedstawiono implementację algorytmu Rungego-Kutty IV rzędu (RK4) oraz RK45 (IV rzędu z krokiem korygowanym rzędem V) oraz wyniki symulacji modelu matematycznego sieci interakcji biochemicznych w czterech scenariuszach stanu komórki.

2. Opis zaimplementowanego rozwiązania

- **Wariant implementacji:** RK4 ze stałym krokiem oraz RK45 ze zmiennym krokiem
- **Biblioteki użyte:** numpy, matplotlib
- **Model:** Symulacja dynamiki białek p53, MDMcyto, MDMn, PTEN w czterech scenariuszach biologicznych
- **Horyzont czasowy:** 48 godzin (2880 minut)
- **Krok całkowania:** 0.5 minuty
- **Wprowadzanie danych:** Ręczne w terminalu lub przyjęcie domyślnych

Źródła przyjętych stężeń

1. **p53**

- **Stężenie:** ~10–100 nM (w komórkach niepoddanych stresowi; silnie zmienne w zależności od typu komórki i warunków)
- **Literatura:**
  - Purvis et al. (2012), "*p53 dynamics control cell fate.*" *Science, 336(6087), 1440-1444.* DOI: 10.1126/science.1218351
  - Batchelor et al. (2011), "*Stimulus-dependent dynamics of p53 in single cells.*" *Molecular Systems Biology, 7, 488.* DOI: 10.1038/msb.2011.20

2. **MDM2 (MDMcyto – cytoplazmatyczny MDM2)**

- **Stężenie:** ~50–200 nM (wyższe w komórkach nowotworowych)
- **Literatura:**
  - Lahav et al. (2004), "*Dynamics of the p53-MDM2 feedback loop in individual cells.*" *Nature Genetics, 36(2), 147-150.* DOI: 10.1038/ng1293

3. **MDM2 (MDMn – jądrowy MDM2)**

- **Stężenie:** Podobne do cytoplazmatycznego MDM2 (~50–200 nM), ale poziomy jądrowe mogą się wahać w wyniku regulacji przez p53
- **Literatura:**
  - Geva-Zatorsky et al. (2006), "*Oscillations and variability in the p53 system.*" *Molecular Systems Biology, 2, 2006.0033.* DOI: 10.1038/msb4100068

4. PTEN

- **Stężenie:** ~100–500 nM (różni się w zależności od tkanki; niższe w niektórych nowotworach z powodu haploinsuficjencji)
- **Literatura:**
  - Leslie et al. (2008), "*The importance of PTEN phosphorylation in vivo.*" *Biochemical Society Transactions, 36(Pt 3), 287-291.* DOI: 10.1042/BST0360287
  - Alimonti et al. (2010), "*Subtle variations in PTEN dose determine cancer susceptibility.*" *Nature Genetics, 42(5), 454-458.* DOI: 10.1038/ng.556

Uwagi:

- Stężenia mogą się znacznie różnić między typami komórek (np. nowotworowe vs. prawidłowe).
- Poziomy p53 są ściśle regulowane przez degradację zależną od MDM2 i wykazują dynamiczne fluktuacje.
- Poziomy PTEN są często obniżone w nowotworach z powodu utraty jednej kopii genu.
- Kierując się tą wiedzą zalecane byłoby stosowanie różnych warunków początkowych w poszczególnych scenariuszach.

Przeliczenie liczby cząsteczek białek w hepatocycie

Obie moje publikacje naukowe dotyczą wątroby, więc wybrałem sobie, że będę modelował hepatocyt.

Założenia:

- Objętość hepatocytu: **3,4 × 10^3 µm^3** = **3,4 × 10^{-12} L** (1 µm^3 = 10^{-15} L)
- Liczba Avogadra: **6,022 × 10^{23} cząsteczek/mol**
- Stężenia przyjęte (środki przedziałów):
  - **p53:** 50 nM = 50 × 10^{-9} mol/L
  - **MDMcyto:** 100 nM = 100 × 10^{-9} mol/L
  - **MDMn:** 100 nM = 100 × 10^{-9} mol/L
  - **PTEN:** 300 nM = 300 × 10^{-9} mol/L

Obliczenia dla każdego białka:

1. **p53**
   - Liczba moli:
     $n = 50 \times 10^{-9}\ \text{mol/L} \times 3{,}4 \times 10^{-12}\ \text{L} = 1{,}7 \times 10^{-19}\ \text{mol}$

- Liczba cząsteczek:
  $N = n \times N_A = 1{,}7 \times 10^{-19} \times 6{,}022 \times 10^{23} \approx 102\,374$

2. **MDMcyto**

   - Liczba moli:
     $n = 100 \times 10^{-9} \times 3{,}4 \times 10^{-12} = 3{,}4 \times 10^{-19}$
   - Liczba cząsteczek:
     $N = 3{,}4 \times 10^{-19} \times 6{,}022 \times 10^{23} \approx 204\,749$

3. **MDMn**

   - Analogicznie jak wyżej:
     $N \approx 204\,749$

4. **PTEN**

   - Liczba moli:
     $n = 300 \times 10^{-9} \times 3{,}4 \times 10^{-12} = 1{,}02 \times 10^{-18}$
   - Liczba cząsteczek:
     $N = 1{,}02 \times 10^{-18} \times 6{,}022 \times 10^{23} \approx 614\,247$

Podsumowanie (zaokrąglone):

Białko	Stężenie [nM]	Liczba cząsteczek w hepatocycie
p53	50	~102 000
MDMcyto	100	~205 000
MDMn	100	~205 000
PTEN	300	~614 000

Wzór użyty do obliczeń:
$N = C \times V \times N_A$
gdzie:

- C – stężenie (mol/L)
- V – objętość komórki (L)
- N_A – liczba Avogadra ($6{,}022 \times 10^{23}$ cząsteczek/mol)

Źródło objętości komórki:
Naryzhny, S. (2023). Quantitative aspects of the human cell Proteome. Int J Mol Sci, 24(10), 8524.

In [2]:
```python
# Importy
import numpy as np
import matplotlib.pyplot as plt

# Parametry modelu
p1 = 8.8
p2 = 440
p3 = 100
d1 = 1.375*(10**-14)
d2 = 1.375*(10**-4)
```

```
d3 = 3*(10**-5)
k1 = 1.925*(10**-4)
k2 = 10**5
k3 = 1.5*(10**5)
value_siRNA = 0.02
value_PTEN_off = 0
value_no_DNA_damage = 0.1

def f_p53(p53, mdmn):
    return p1 - d1*p53*(mdmn**2)

def f_mdmcyto(p53, mdmcyto, pten, siRNA=False, no_DNA_damage=False):
    siRNA_factor = value_siRNA if siRNA else 1
    DNA_damage_factor = value_no_DNA_damage if no_DNA_damage else 1
    return p2*siRNA_factor*(p53**4)/((p53**4) + (k2**4)) - k1*(k3**2)/((k3**2) +

def f_mdmn(mdmn, mdmcyto, pten, no_DNA_damage=False):
    if no_DNA_damage:
        return k1*(k3**2)/((k3**2) + (pten**2))*mdmcyto - d2*value_no_DNA_damage
    else:
        return k1*(k3**2)/((k3**2) + (pten**2))*mdmcyto - d2*mdmn

def f_pten(pten, p53, pten_off=False):
    if not pten_off:
        return p3*(p53**4)/((p53**4) + (k2**4)) - d3*pten
    else:
        return p3*value_PTEN_off*(p53**4)/((p53**4) + (k2**4)) - d3*pten
```

Tu nie ma wiele do wyjaśnienia. Anna napisała to po mistrzowsku i jest przejrzyście i jasno zaimplementowane.

In [3]:
```
# RK4 ze stałym krokiem
def RK4const(p53, mdcyto, mdmn, pten, h, siRNA=False, pten_off=False, no_DNA_dam
    k1_p53 = f_p53(p53, mdmn)
    k1_mdmcyto = f_mdmcyto(p53, mdcyto, pten, siRNA)
    k1_mdmn = f_mdmn(mdmn, mdcyto, pten)
    k1_pten = f_pten(pten, p53, pten_off)

    k2_p53 = f_p53(p53 + h/2*k1_p53, mdmn + h/2*k1_mdmn)
    k2_mdmcyto = f_mdmcyto(p53 + h/2*k1_p53, mdcyto + h/2*k1_mdmcyto, pten + h/2
    k2_mdmn = f_mdmn(mdmn + h/2*k1_mdmn, mdcyto + h/2*k1_mdmcyto, pten + h/2*k1_
    k2_pten = f_pten(pten + h/2*k1_pten, p53 + h/2*k1_p53, pten_off)

    k3_p53 = f_p53(p53 + h/2*k2_p53, mdmn + h/2*k2_mdmn)
    k3_mdmcyto = f_mdmcyto(p53 + h/2*k2_p53, mdcyto + h/2*k2_mdmcyto, pten + h/2
    k3_mdmn = f_mdmn(mdmn + h/2*k2_mdmn, mdcyto + h/2*k2_mdmcyto, pten + h/2*k2_
    k3_pten = f_pten(pten + h/2*k2_pten, p53 + h/2*k2_p53, pten_off)

    k4_p53 = f_p53(p53 + h*k3_p53, mdmn + h*k3_mdmn)
    k4_mdmcyto = f_mdmcyto(p53 + h*k3_p53, mdcyto + h*k3_mdmcyto, pten + h*k3_pt
    k4_mdmn = f_mdmn(mdmn + h*k3_mdmn, mdcyto + h*k3_mdmcyto, pten + h*k3_pten)
    k4_pten = f_pten(pten + h*k3_pten, p53 + h*k3_p53,  pten_off)

    p53 += (k1_p53 + 2*k2_p53 + 2*k3_p53 + k4_p53) * h / 6
    mdcyto += (k1_mdmcyto + 2*k2_mdmcyto + 2*k3_mdmcyto + k4_mdmcyto) * h / 6
    mdmn += (k1_mdmn + 2*k2_mdmn + 2*k3_mdmn + k4_mdmn) * h / 6
    pten += (k1_pten + 2*k2_pten + 2*k3_pten + k4_pten) * h / 6
    return p53, mdcyto, mdmn, pten
```

To moje autorskie rozwiązanie, krok jest modyfikowany aż do skutku.

In [4]:
```
# RK45 ze zmiennym krokiem
def RK45(p53, mdmcyto, mdmn, pten, h, siRNA=False, pten_off=False, no_DNA_damage
    def f(t, y):
        yp53 = f_p53(y[0], y[2])
        ymdmcyto = f_mdmcyto(y[0], y[1], y[3], siRNA, no_DNA_damage)
        ymdmn = f_mdmn(y[2], y[1], y[3], no_DNA_damage)
        ypten = f_pten(y[3], y[0], pten_off)
        return np.array([yp53, ymdmcyto, ymdmn, ypten])

    y = np.array([p53, mdmcyto, mdmn, pten], dtype=float)
    t = 0  # dummy, as system is autonomous

    while True:
        # Skorzystaliśmy z tablicy dla R-K 45-Dormand-Prince (z Pańskiego wykład
        k1 = f(t, y)
        k2 = f(t + h*(1/5), y + h*(1/5)*k1)
        k3 = f(t + h*(3/10), y + h*(3/40*k1 + 9/40*k2))
        k4 = f(t + h*(4/5), y + h*(44/45*k1 - 56/15*k2 + 32/9*k3))
        k5 = f(t + h*(8/9), y + h*(19372/6561*k1 - 25360/2187*k2 + 64448/6561*k3
        k6 = f(t + h, y + h*(9017/3168*k1 - 355/33*k2 + 46732/5247*k3 + 49/176*k
        k7 = f(t + h, y + h*(35/384*k1 + 500/1113*k3 + 125/192*k4 - 2187/6784*k5

        y5 = y + h*(35/384*k1 + 500/1113*k3 + 125/192*k4 - 2187/6784*k5 + 11/84*
        y4 = y + h*(5179/57600*k1 + 7571/16695*k3 + 393/640*k4 - 92097/339200*k5

        err = np.linalg.norm(y5 - y4)
        tol = atol + rtol * np.linalg.norm(y5)

        if err <= tol or h < 1e-8:
            return tuple(y5)
        if err == 0:
            s = 2
        else:
            s = 0.9 * (tol / err)**(1/5)
        h *= min(max(0.1, s), 5.0)
```

SPB - czesc 2.pdf Zamknij

Zmienny krok w metodach Runge-Kutty

R-K 45 – Dormand-Prince:

0							
1/5	1/5						
3/10	3/40	9/40					
4/5	44/45	–56/15	32/9				
8/9	19372/6561	–25360/2187	64448/6561	–212/729			
1	9017/3168	–355/33	46732/5247	49/176	–5103/18656		
1	35/384	0	500/1113	125/192	–2187/6784	11/84	
	5179/57600	0	7571/16695	393/640	–92097/339200	187/2100	1/40
	35/384	0	500/1113	125/192	–2187/6784	11/84	0

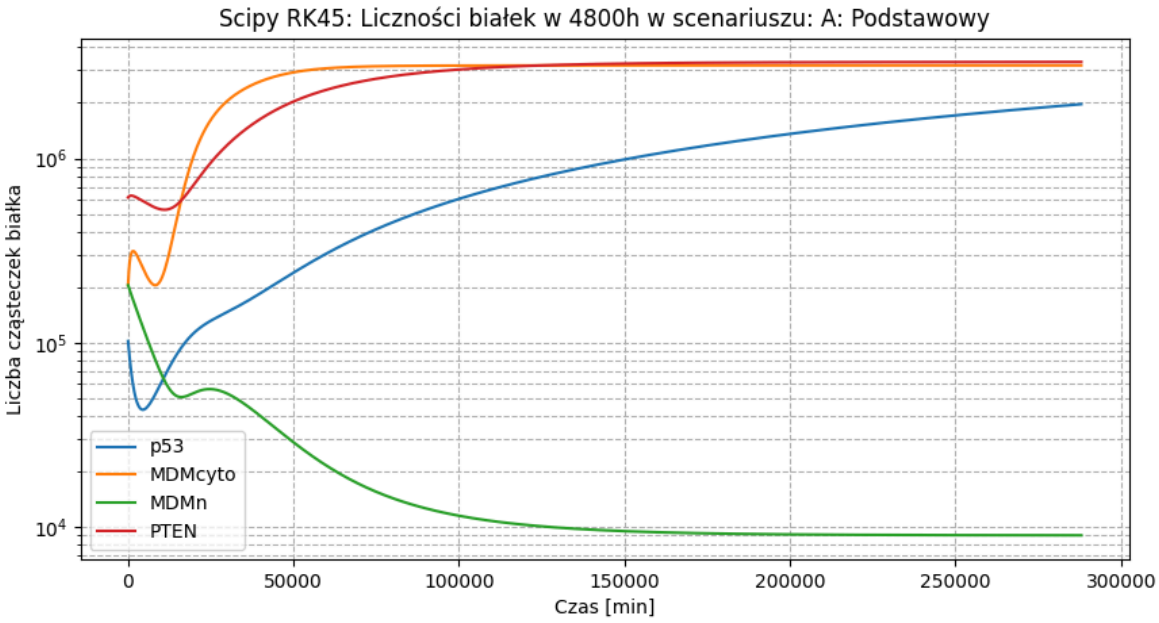
15 / 15

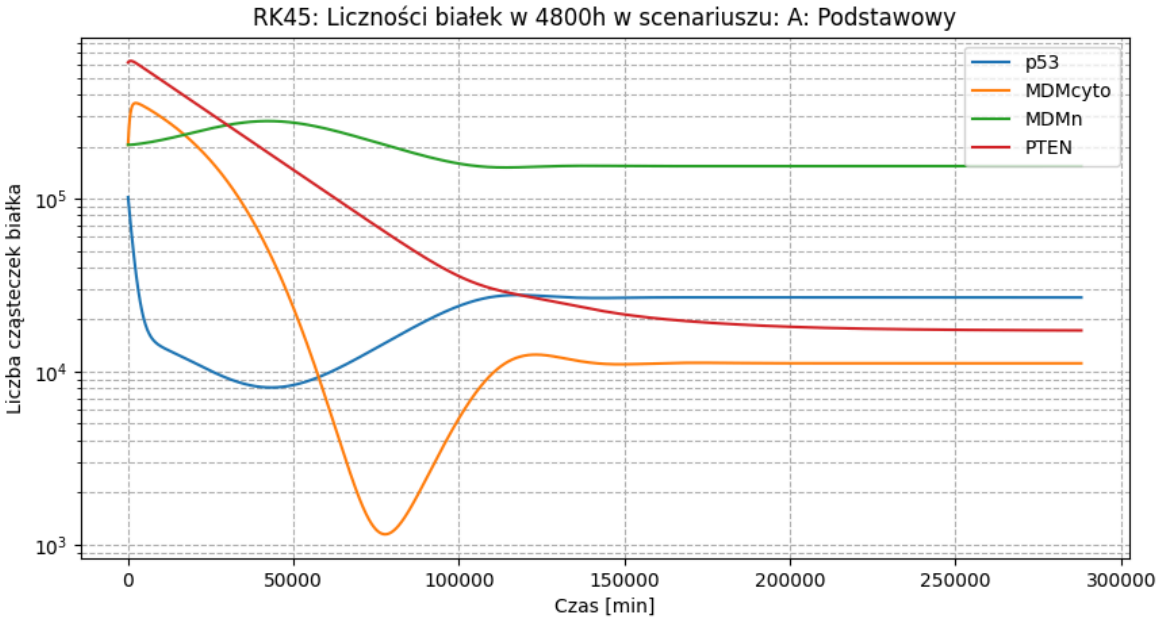
3. Parametry symulacji i scenariusze

Najpierw chcemy zobaczyć, jak wygląda wykres scenariusza podstawowego dla liczności białek policzonych we wporwadzeniu.

In [6]:
```
# Warunki początkowe zgodnie z liczbą cząsteczek w hepatocycie:
p53_0 = 102000
mdmcyto_0 = 205000
mdmn_0 = 205000
pten_0 = 614000
h = 0.5 # krok czasowy w minutach
iterations = int(4800*60/h) # liczba iteracji na 4800h

conditions = {
    "A: Podstawowy": (False, False, True),  # brak siRNA, PTEN działa, brak uszk
}

scenarios = list(conditions.keys())
# RK4const
for scenario in scenarios:
    siRNA, pten_off, no_DNA_damage = conditions[scenario]
    p53, mdmcyto, mdmn, pten = p53_0, mdmcyto_0, mdmn_0, pten_0
    time_values = []
    p53_values = []
    mdmcyto_values = []
    mdmn_values = []
    pten_values = []
    for i in range(iterations):
        time = i * h
        time_values.append(time)
        p53_values.append(p53)
        mdmcyto_values.append(mdmcyto)
        mdmn_values.append(mdmn)
        pten_values.append(pten)
        p53, mdmcyto, mdmn, pten = RK4const(p53, mdmcyto, mdmn, pten, h, siRNA,
    plt.figure(figsize=(10,5))
    plt.plot(time_values, p53_values, label="p53")
    plt.plot(time_values, mdmcyto_values, label="MDMcyto")
    plt.plot(time_values, mdmn_values, label="MDMn")
    plt.plot(time_values, pten_values, label="PTEN")
    plt.xlabel("Czas [min]")
    plt.ylabel("Liczba cząsteczek białka")
    plt.title(f"Scipy RK45: Liczności białek w 4800h w scenariuszu: {scenario}")
    plt.yscale("log")
    plt.legend()
    plt.grid(True, which="both", ls="--")
    plt.show()

p53_0 = 102000
mdmcyto_0 = 205000
mdmn_0 = 205000
pten_0 = 614000
h = 0.5 # krok czasowy w minutach
iterations = int(4800*60/h) # liczba iteracji na 4800h

conditions = {
    "A: Podstawowy": (False, False, True),  # brak siRNA, PTEN działa, brak uszk
```

```
}

scenarios = list(conditions.keys())

# RK45
for scenario in scenarios:
    siRNA, pten_off, no_DNA_damage = conditions[scenario]
    p53, mdmcyto, mdmn, pten = p53_0, mdmcyto_0, mdmn_0, pten_0
    time_values = []
    p53_values = []
    mdmcyto_values = []
    mdmn_values = []
    pten_values = []
    for i in range(iterations):
        time = i * h
        time_values.append(time)
        p53_values.append(p53)
        mdmcyto_values.append(mdmcyto)
        mdmn_values.append(mdmn)
        pten_values.append(pten)
        p53, mdmcyto, mdmn, pten = RK45(p53, mdmcyto, mdmn, pten, h, siRNA=siRNA
    plt.figure(figsize=(10,5))
    plt.plot(time_values, p53_values, label="p53")
    plt.plot(time_values, mdmcyto_values, label="MDMcyto")
    plt.plot(time_values, mdmn_values, label="MDMn")
    plt.plot(time_values, pten_values, label="PTEN")
    plt.xlabel("Czas [min]")
    plt.ylabel("Liczba cząsteczek białka")
    plt.title(f"RK45: Liczności białek w 4800h w scenariuszu: {scenario}")
    plt.yscale("log")
    plt.legend()
    plt.grid(True, which="both", ls="--")
    plt.show()
```





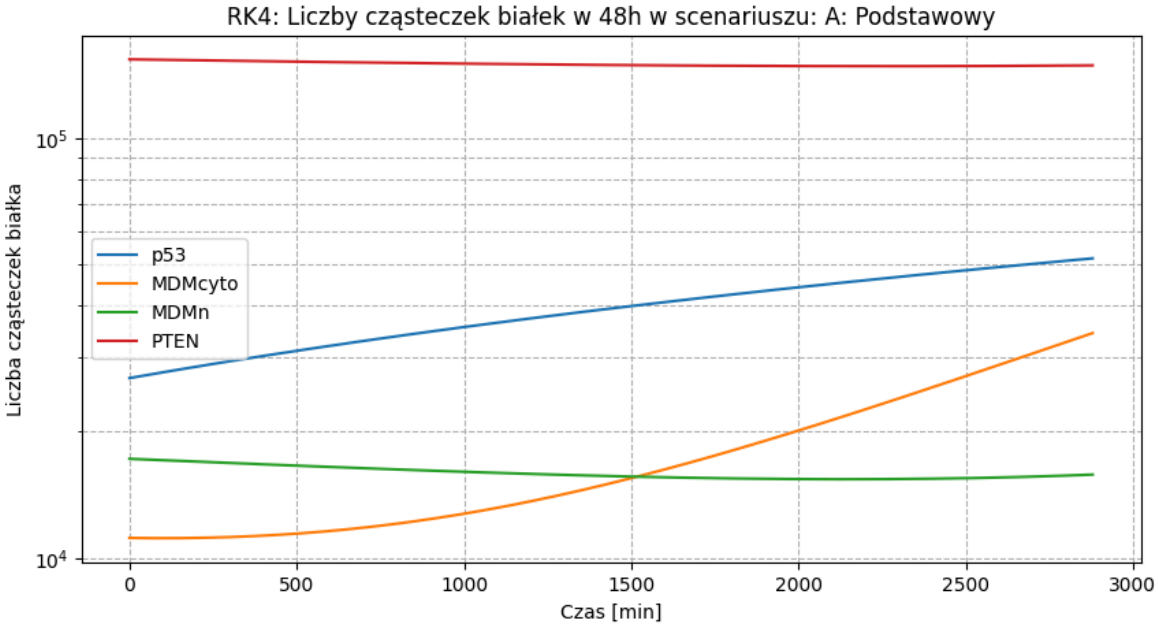
Potem liczności utrzymujące się po stabilizacji modelu wykorzystujemy jako dane wejściowe do wszystkich czterech scenariuszy:

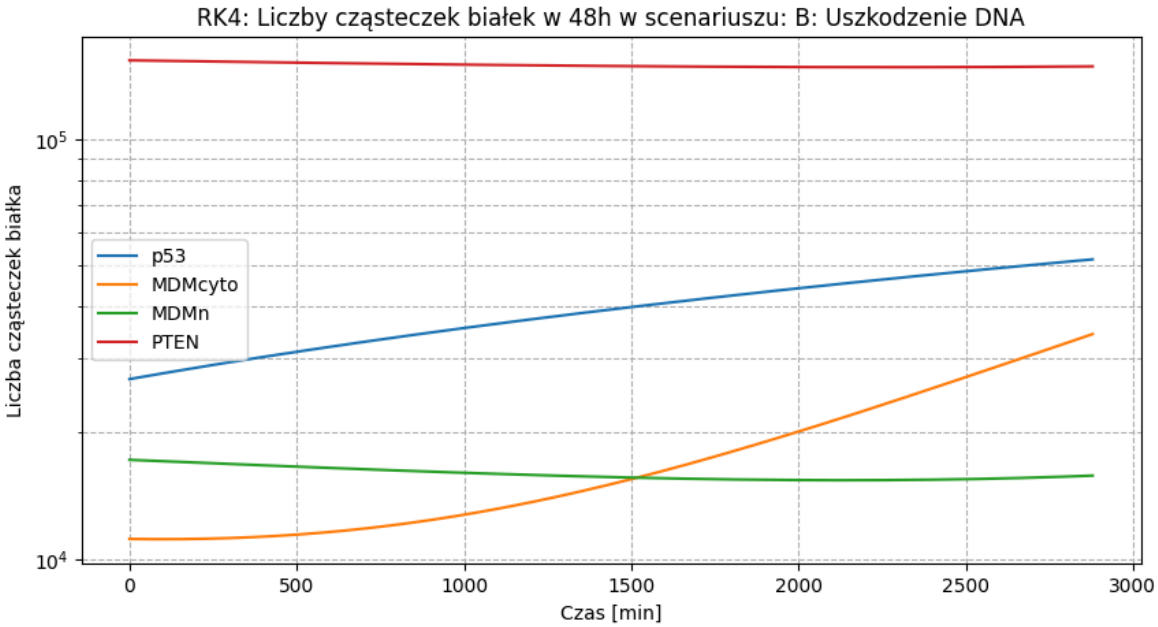
In [7]:
```
# Parametry symulacji
h = 0.5 # krok w minutach
iterations = int(48*60/h) # liczba iteracji na 48h
p53_0 = 26854
mdmcyto_0 = 11173
mdmn_0 = 17245
pten_0 = 154378

conditions = {
    "A: Podstawowy": (False, False, True),  # brak siRNA, PTEN działa, brak uszk
    "B: Uszkodzenie DNA": (False, False, False),  # brak siRNA, PTEN działa, usz
    "C: Nowotwór": (False, True, False),  # brak siRNA, PTEN wyłączony, uszkodze
    "D: Terapia": (True, True, False),  # siRNA, PTEN wyłączony, uszkodzenie DNA
}

scenarios = list(conditions.keys())
```

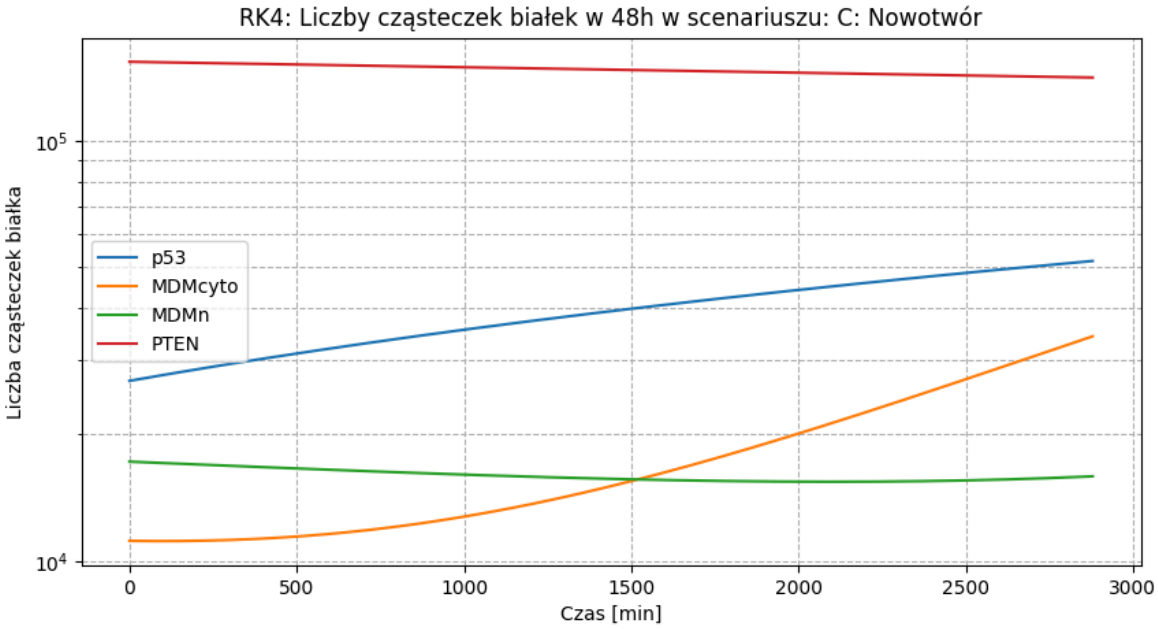
4. Wyniki symulacji: RK4 (stały krok)

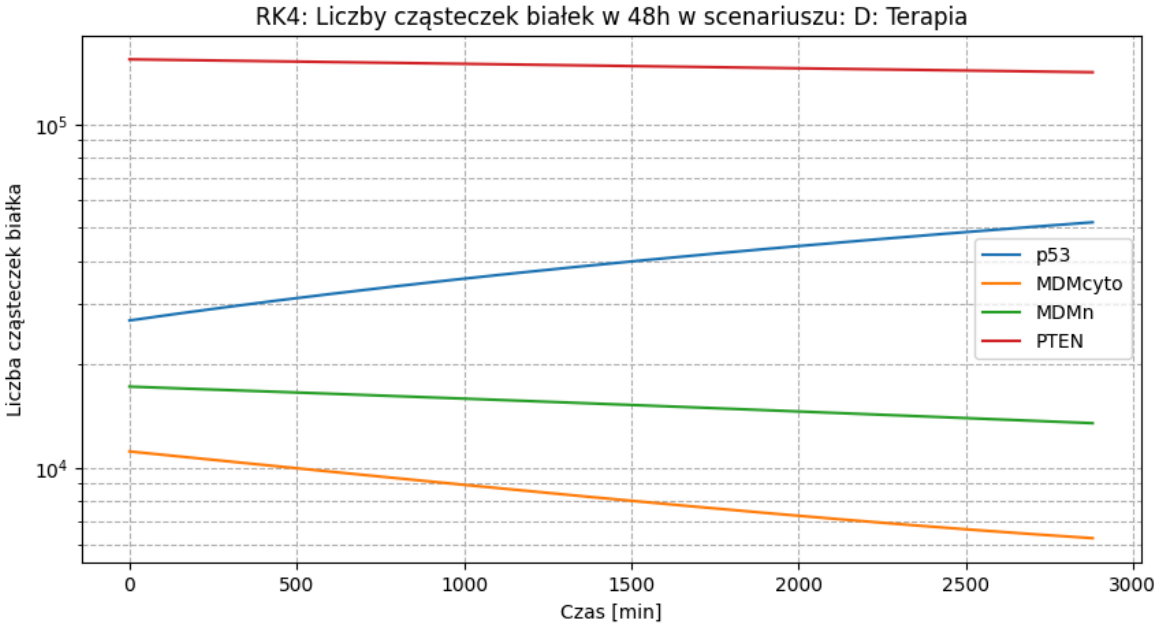
In [8]:
```
for scenario in scenarios:
    siRNA, pten_off, no_DNA_damage = conditions[scenario]
    p53, mdmcyto, mdmn, pten = p53_0, mdmcyto_0, mdmn_0, pten_0
    time_values = []
    p53_values = []
    mdmcyto_values = []
    mdmn_values = []
    pten_values = []
    for i in range(iterations):
        time = i * h
        time_values.append(time)
        p53_values.append(p53)
        mdmcyto_values.append(mdmcyto)
        mdmn_values.append(mdmn)
        pten_values.append(pten)
        p53, mdmcyto, mdmn, pten = RK4const(p53, mdmcyto, mdmn, pten, h, siRNA,
```

```
    plt.figure(figsize=(10,5))
    plt.plot(time_values, p53_values, label="p53")
    plt.plot(time_values, mdmcyto_values, label="MDMcyto")
    plt.plot(time_values, mdmn_values, label="MDMn")
    plt.plot(time_values, pten_values, label="PTEN")
    plt.xlabel("Czas [min]")
    plt.ylabel("Liczba cząsteczek białka")
    plt.title(f"RK4: Liczby cząsteczek białek w 48h w scenariuszu: {scenario}")
    plt.yscale("log")
    plt.legend()
    plt.grid(True, which="both", ls="--")
    plt.show()
```







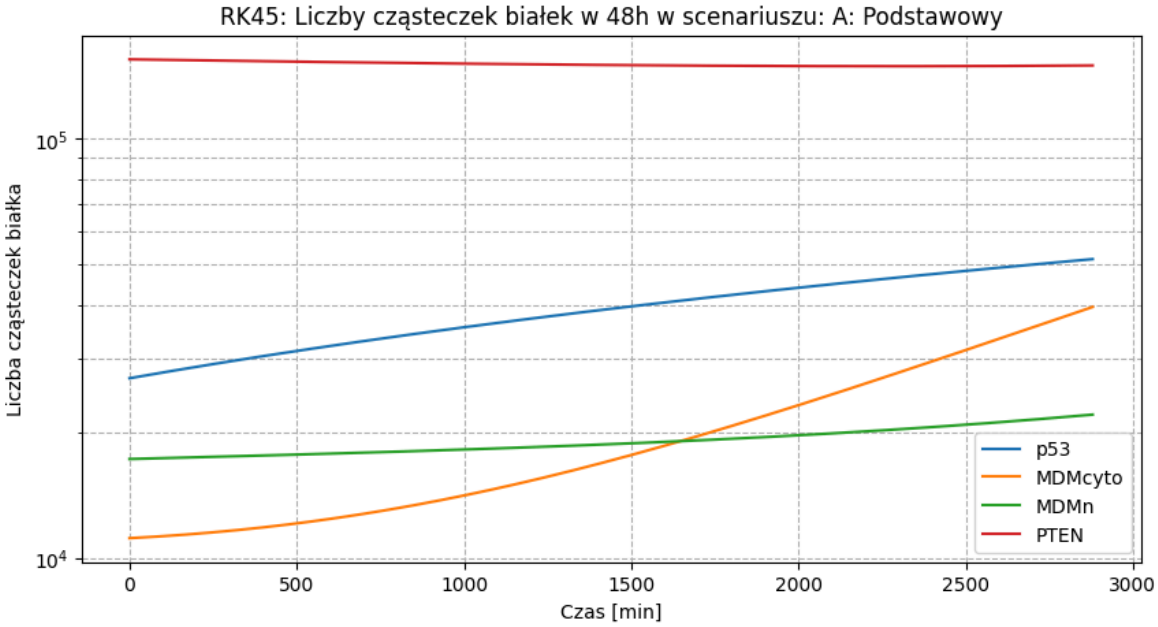


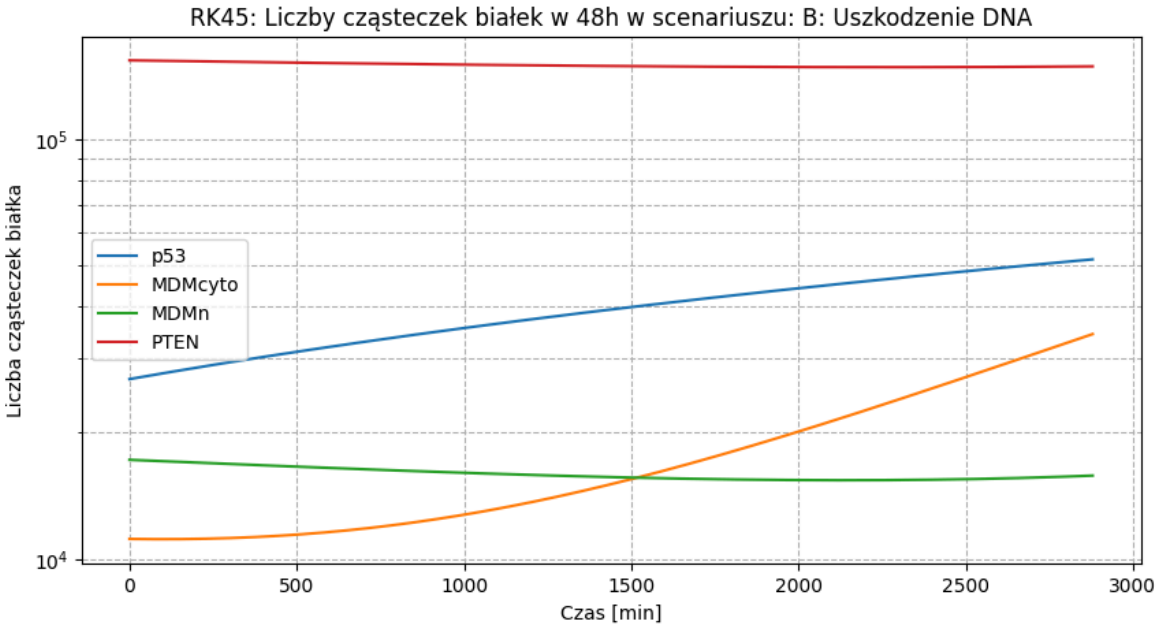
5. Wyniki symulacji: RK45 (adaptacyjny krok)

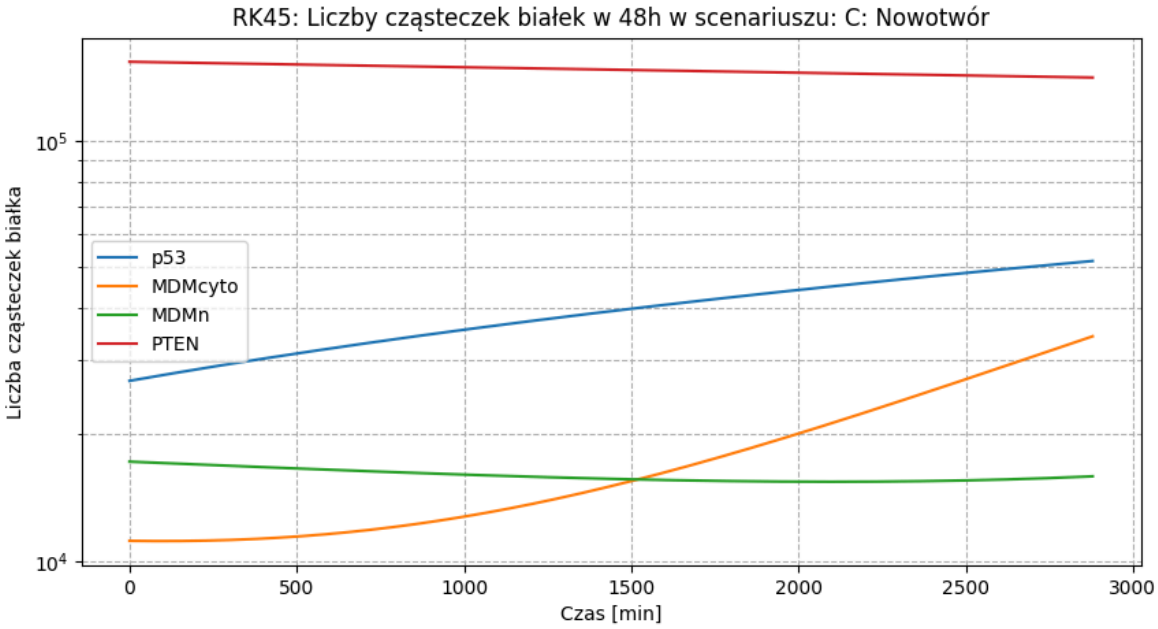
In [9]:

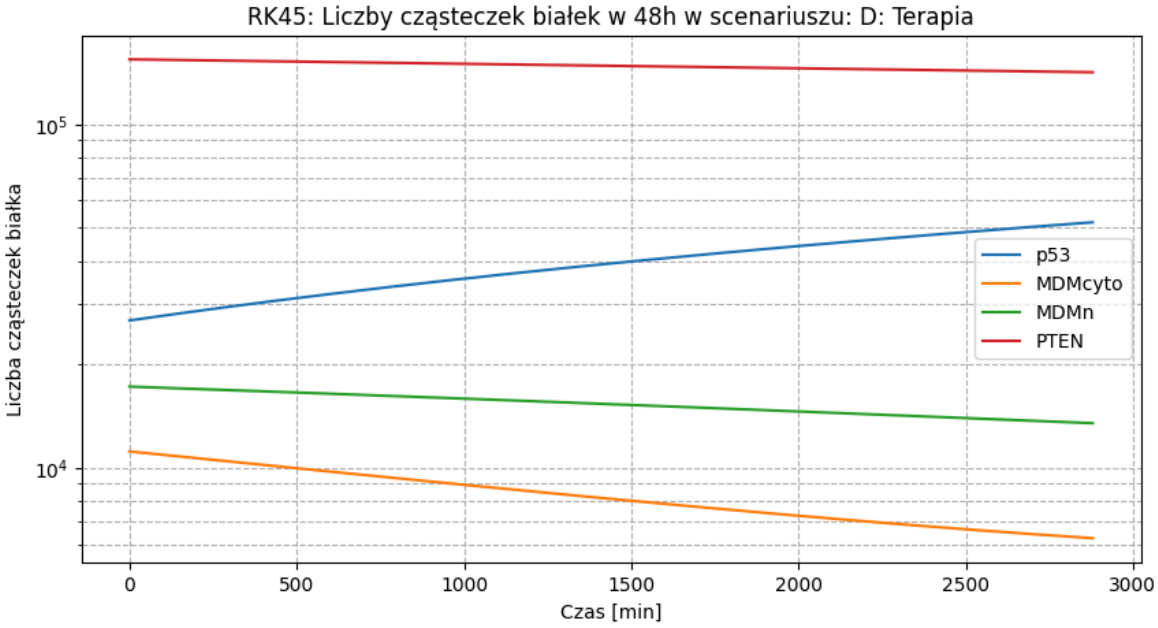
```
for scenario in scenarios:
    siRNA, pten_off, no_DNA_damage = conditions[scenario]
    p53, mdmcyto, mdmn, pten = p53_0, mdmcyto_0, mdmn_0, pten_0
    time_values = []
    p53_values = []
    mdmcyto_values = []
    mdmn_values = []
    pten_values = []
    for i in range(iterations):
        time = i * h
        time_values.append(time)
        p53_values.append(p53)
        mdmcyto_values.append(mdmcyto)
        mdmn_values.append(mdmn)
        pten_values.append(pten)
        p53, mdmcyto, mdmn, pten = RK45(p53, mdmcyto, mdmn, pten, h, siRNA=siRNA
    plt.figure(figsize=(10,5))
```

```
    plt.plot(time_values, p53_values, label="p53")
    plt.plot(time_values, mdmcyto_values, label="MDMcyto")
    plt.plot(time_values, mdmn_values, label="MDMn")
    plt.plot(time_values, pten_values, label="PTEN")
    plt.xlabel("Czas [min]")
    plt.ylabel("Liczba cząsteczek białka")
    plt.title(f"RK45: Liczby cząsteczek białek w 48h w scenariuszu: {scenario}")
    plt.yscale("log")
    plt.legend()
    plt.grid(True, which="both", ls="--")
    plt.show()
```





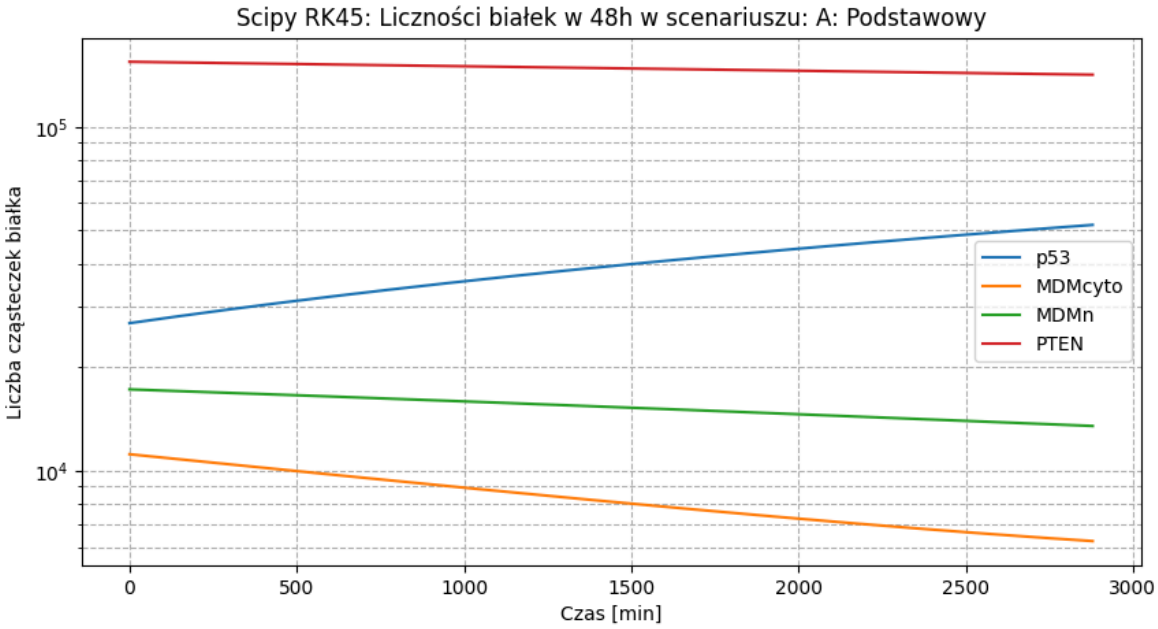


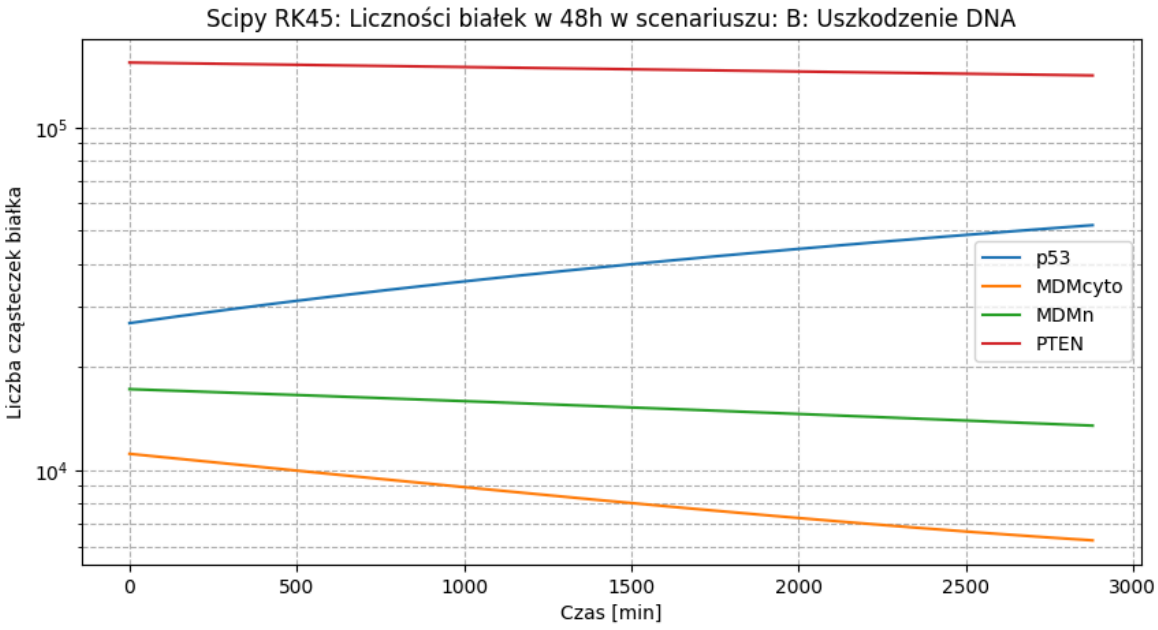


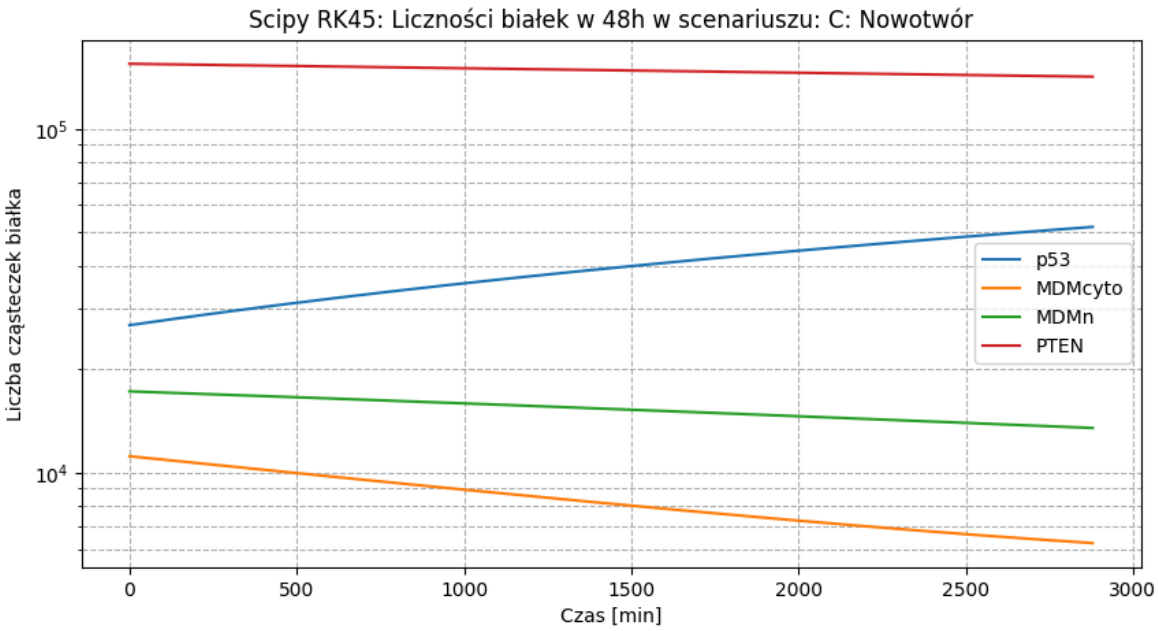
Dla porównania pozwolę sobie skorzystać z rozwiązań gotowych oferowanych przez pakiet Scipy (niby nie mówił Profesor, że nie można korzystać z takich, ale uznaliśmy że nie wypada, więc to tylko kontorlnie):

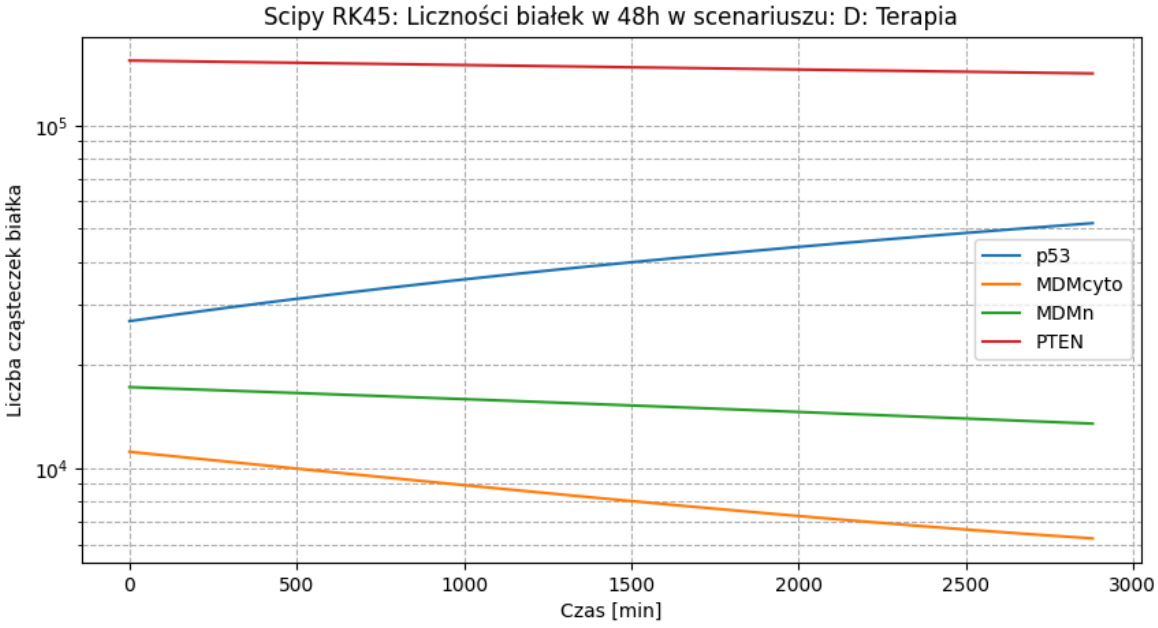
In [11]:
```
from scipy.integrate import solve_ivp
def model_ode(t, y, siRNA=False, pten_off=False, no_DNA_damage=False):
    p53, mdmcyto, mdmn, pten = y
    dydt = np.zeros(4)
    dydt[0] = f_p53(p53, mdmn)
    dydt[1] = f_mdmcyto(p53, mdmcyto, pten, siRNA=siRNA, no_DNA_damage=no_DNA_da
    dydt[2] = f_mdmn(mdmn, mdmcyto, pten, no_DNA_damage=no_DNA_damage)
    dydt[3] = f_pten(pten, p53, pten_off=pten_off)
    return dydt
for scenario in scenarios:
    siRNA, pten_off, no_DNA_damage = conditions[scenario]
    # Definicja zakresu czasu i warunków początkowych
    t_span = (0, 48*60)  # od 0 do 2880 minut (48h)
    y0 = [p53_0, mdmcyto_0, mdmn_0, pten_0]
    t_eval = np.arange(0, 48*60 + h, h)
    sol = solve_ivp(
```

```
        model_ode, t_span, y0, method='RK45', t_eval=t_eval,
        args=(siRNA, pten_off, no_DNA_damage)
    )
    plt.figure(figsize=(10,5))
    plt.plot(time_values, p53_values, label="p53")
    plt.plot(time_values, mdmcyto_values, label="MDMcyto")
    plt.plot(time_values, mdmn_values, label="MDMn")
    plt.plot(time_values, pten_values, label="PTEN")
    plt.xlabel("Czas [min]")
    plt.ylabel("Liczba cząsteczek białka")
    plt.title(f"Scipy RK45: Liczności białek w 48h w scenariuszu: {scenario}")
    plt.yscale("log")
    plt.legend()
    plt.grid(True, which="both", ls="--")
    plt.show()
```









6. Wnioski

1. Wszędzie za wyjątkiem scenariusza A symulowanego algorytmem z krokiem zmiennym widzimy spadek MDM
2. Druga znaczna różnica w wynikach objawia się w scenariuszu D) Terapia – wyłączony PTEN, załączone uszkodzenia DNA i załączone siRNA.
3. Pozostałe różnice między scenariuszami dla przyjętych danych są kosmetyczne - warto by było gdyby profesor dał jakieś dane do wytestowania