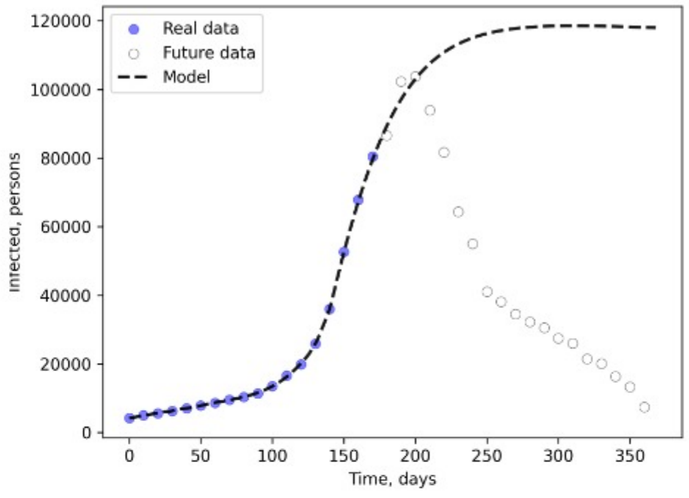
Первый запуск: берем первоначальный loss_dinn и просим изменить лосс под соответствующий комментарий

```python
def loss_dinn(S_hat, S_pred, I_hat, I_pred, D_hat, D_pred, R_hat, R_pred, f1, f2, f3, f4, I_pred_last):

    regul = 0.8
    last_infected_penalty = 0.05

    aggregation_func = torch.mean
    norm_func = torch.square

    term1 = aggregation_func(norm_func(S_hat - S_pred))
    term2 = aggregation_func(norm_func(I_hat - I_pred))
    term3 = aggregation_func(norm_func(D_hat - D_pred))
    term4 = aggregation_func(norm_func(R_hat - R_pred))

    term5 = aggregation_func(norm_func(f1))
    term6 = aggregation_func(norm_func(f2))
    term7 = aggregation_func(norm_func(f3))
    term8 = aggregation_func(norm_func(f4))

    loss = regul * (term1 + term2 + term3 + term4) + \
        (1 - regul) * (term5 + term6 + term7 + term8) + \
        last_infected_penalty * norm_func(I_pred_last-0)
    return loss
```

loss_dinn_primary.py

Comment:"The infection rate in the forecast is too high, given the vaccination rate."

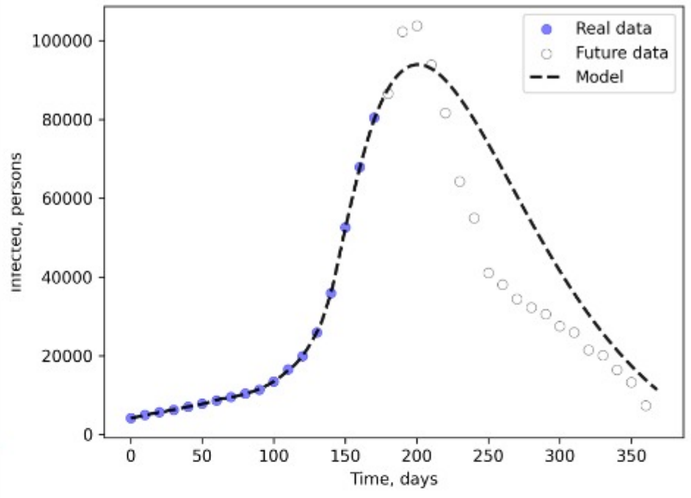
Второй запуск: берем нагенеренный loss_dinn и просим изменить лосс под соответствующий комментарий




```python
def loss_dinn(S_hat, S_pred, I_hat, I_pred, D_hat, D_pred, R_hat, R_pred, f1, f2, f3, f4, I_pred_last):
    regul = 0.8
    last_infected_penalty = 0.05
    aggregation_func = torch.mean
    norm_func = torch.square

    term1 = aggregation_func(norm_func(S_hat - S_pred))
    term2 = aggregation_func(norm_func(I_hat - I_pred))
    #term3 = aggregation_func(norm_func(D_hat - D_pred))
    #term4 = aggregation_func(norm_func(R_hat - R_pred))
    term5 = aggregation_func(norm_func(f1))
    term6 = aggregation_func(norm_func(f2))
    #term7 = aggregation_func(norm_func(f3))
    #term8 = aggregation_func(norm_func(f4))

    loss = regul * (term1 + term2) + (1 - regul) * (term5 + term6)
    return loss
```

Comment:"In 20 days the number of infected should start to decrease."

Результат: график кол-ва инфицированных и окончательная функция loss_dinn




```python
import torch


def loss_dinn(S_hat, S_pred, I_hat, I_pred, D_hat, D_pred, R_hat, R_pred, f1, f2, f3, f4, I_pred_last):
    regul = 0.8
    last_infected_penalty = 0.05
    aggregation_func = torch.mean
    norm_func = torch.square

    term1 = aggregation_func(norm_func(S_hat - S_pred))
    term2 = aggregation_func(norm_func(I_hat - I_pred)) + (I_pred_last - I_hat).abs().mean() * last_infected_penalty  # added pe
    #term3 = aggregation_func(norm_func(D_hat - D_pred))
    #term4 = aggregation_func(norm_func(R_hat - R_pred))
    term5 = aggregation_func(norm_func(f1))
    term6 = aggregation_func(norm_func(f2))

    loss = regul * (term1 + term2) + (1 - regul) * (term5 + term6)
    return loss
```

loss_dinn_LLM.py