20/11/25 10am (Session 1-2)

Summary of what I've taken from the introductory slides:

What why when:

-> using existing PIC code written in Python, conduct plasma physics study.

-> various tasks on VLE, broken into separate sections

-> Sessions run flexibly, don't need to be in the room, don't need to be in front of computer all day

-> ~28 hrs of work ish (30 max)

Assessment:

- google doc lab book
- pass/fail
- assessment based entirely off lab book, submitted through VLE as pdf
- much better to submit a succinct, high quality record, rather than a verbose, low quality one.

Quick start guide:

- implements PIC algorithm in 1D periodic domain, using RK4 for time integration and FFTs for spatial derivatives (SEE LECTURES 7 AND 8 OF COMP TECHNIQUES!!!!!)
- primary use of code = look at wave-particle interactions
- Two main physics phenomena explored in the tasks
    - Landau damping: transfer of Energy from wave to plasma
    - two stream instability: transfer of Energy from plasma to wave

Tips:

- code can be quite slow, can you speed it up???
- DONT FORGET ERRORS AND UNCERTAINTIES!!!
    - PIC suffers from noise (t.f. may need lots of repeats which may generate lots of data).
    - How should I handle this???
- Split into 2 stages:
    - generation of raw data by running code -> save raw data
    - analyse that raw data. i.e. measure derived quantities from save draw data using anal. code.

First Steps:

- review tasks

- read up a lil on wave-particle interactions

    - produce summary in lab book

- read through code and produce summary of key sections in lab book

    - lecture 8 probs most helpful but lecture 7 can help (comp techniques)

- check you can run code and what the screen output looks like

- produce data management plan

    - GIT!!!!!

- modify code to save the relevant data

    - how will you organise the data?

- make a plan for how you'll tackle the tasks

Data Management Plan

Github Repository is now set up

All coding and analysis will be done using Visual Studio (VS) code, using the source control functionality to backup data and code versions on Github.

Session Plan:

- ✓ Set up github repository and begin lab book entries
- ✓ Download and run code
- ✓ Complete 'Session 1-2: Tasks and Activities'

20/11/25 3pm (Session 1-2)

Summary of code functionality

What is Landau Damping?

- Broadly : A mechanism where oscillations in a charged medium are damped by non-collisional interactions with said medium.
- More specifically, Landau damping occurs due to the energy exchange between and EM wave (with a phase velocity v_{ph}) and particles in the plasma (the charged medium) with a velocity ~ v_{ph}. These particles interact strongly with the wave, as particles with a velocity slightly less than v_{ph} get accelerated by the E-Field of the EM wave, while particles with velocity slightly more than v_{ph} get decelerated. This synchronises the particle velocity with the phase velocity of the wave, which prevents instabilities from developing, creating a region of stability within the parameter space.
- Consider the particle velocities to be approximately Maxwellian – i.e. let the plasma be governed by ideal MHD. If the slope of the function <1, the no. of particles with velocities slightly less than v_{ph} is greater than the no. with velocities slightly greater. This results in wave damping ('Landau Damping'!)as

there are more particles gaining energy from the wave than losing energy to it. AND VICE VERSA -> 'Landau Growth'!

- A nice analogy is the surfing situation:
    - Let Langmuir waves be the sea, and consider the charged particles to be surfers trying to catch the wave.
    - If the surfer is moving at a velocity < v_{ph} , they see the wave approaching from behind, with the wave crest pushing them forward. i.e. they gain energy from the wave, thus the wave loses energy.
    - If the surfer is moving at a velocity > v_{ph} they can be considered as swimming into the crest of the wave, essentially climbing it. i.e. they are losing kinetic energy while the wave is gaining energy.
    - In a Maxwellian distribution, there are more particles with velocities < v_{ph} , thus the slower group dominates and drains energy from the wave, resulting in an overall wave damping effect -> Landau Damping!
    - Worthy to note that only surfers contribute to this effect, as a beachball (for example) floating on the water with zero velocity will simply bob up and down as the wave passes by. In other words, particles very far from v_{ph} oscillate rapidly in and out of phase, so even though they may gain energy in one part of the oscillation cycle, they return it in the next, resulting in a net zero effect over a period.

Initial Experiments with Code

- Simulation runtime with animation : 6.467 seconds
- Simulation run time without animation : 2.110 seconds
- Improvement : 4.357 seconds or 67.372%
- First harmonic data is saved in 2 stack columns (time | first harmonic) as harmonic_data.txt

Assessing the noise

Plan of Action (POA)

- Load or reference harmonic data.
- Detect peaks -> produce arrays of peak times/amps.
- Plot signal + peaks overlay.
- Loop through peaks to find where noise dominates.
- Split data into signal peaks + noise peaks.
- Compute noise metric.
- Print or plot result.

The code according to the POA gives the following results:

--- NOISE ANALYSIS ---
Noise starts at peak index: 3
Estimated noise amplitude: 0.03467561350424121

--- FREQUENCY MEASUREMENT ---
Estimated ω = 2.053 ± 0.137

--- DAMPING RATE ---
Estimated γ = 0.196 ± 0.075

--- COMPARISON ---
Analytic: ω = 1.416, γ = 0.153
Typical PIC: ω = 1.33 ± 0.16, γ = 0.168 ± 0.002

simulation runtime: 8.539 seconds
Initial Analysis:

- Noise dominates after the third peak (fig.1). This gives a signal window that's usable, but short. This can be expected for a low-particle 1D PIC run.
- Later peaks are dominated by particle fluctuations, not damping, as indicated by the noise floor of ~ 3e-2.
- Does the measured ω overshoot analytical/typical PIC values?
- Is γ too high? Is it consistent with analytic and reference PIC values, given the noise-driven uncertainty?
- Longer runs/more particles would tighten estimates and reduce bias.

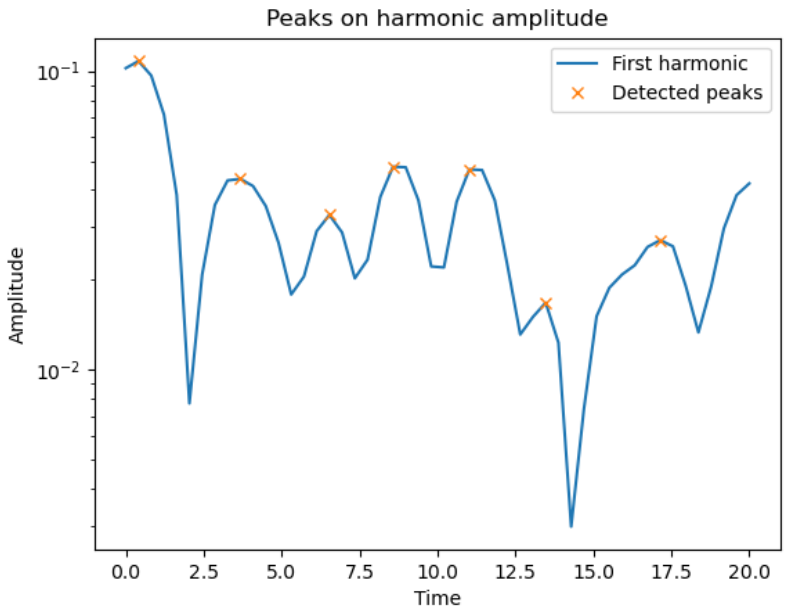


Fig.1 Plot of the First Harmonic Normalised Signal Amplitude with detected peaks overlay. Noise dominates when a peak is larger than the one preceding it. This disagrees with damping, indicating that the signal is being driven by noise.

27/11/25 11pm (Session 2-3)

Goal of the Session

Explore how noise level, wave frequency and damping rate change as number of cells, the number of particles and the length of the box are varied.

Statistical Variation

1. Run 5 sims with same initial conditions
2. Measure the noise, frequency and damping rate for each run.
3. compute a mean with appropriate error estimate.
4. Compare the error on repeat measurements to the error on an individual measurement. Which is most significant?

General idea for code:

1. Ask user for **N runs** (via input() or CLI arg).
2. Loop N times.
3. For each run:
    - run the simulation
    - extract noise, ω, γ
    - save results to a unique file
4. After the loop:
    - compute means + standard deviations
    - print comparison (individual-run error vs amalgamated error)

Code Results:
Noise :- mean=0.03693, std=0.01041
Omega :- mean=2.403, std=0.5554
Gamma :- mean=0.1887, std=0.04362

Error Comparison:
Individual ω error : 0.1642
Run-to-run ω spread: 0.5554
Individual γ error : 0.04336
Run-to-run γ spread: 0.04362

Larger value = dominant source of uncertainty.

27/11/25 2pm (Session 2-3)

Numerical Parameters

How do measurements depend on no. cells and no. markers?

1. Code to time runtime of run method
    a. Done. Wrapped run function with timer.
2. How do the simulation time, noise level, frequency and damping rate vary with number of cells?
    a. Simulation time scales as number of cells multiplied by the log(number of cells) because field solver uses FFT. FFT cost dominates as number of cells grows. Therefore, expect monotonic increase.
    b. For constant total particle number, increasing the number of cells decreases the number of particles per cell, therefore the noise increases : - noise is proportional to square root of ratio of number of cells to number of particles. I.e. More cells means fewer particles per cell, means noisier charge density.
    c. Increasing number of cells refines the spatial grid over which the Langmuir mode frequency is resolved. This gives a better representation of k. So for a coarse grid, there is poor derivative accuracy, and numerical dispersion shifts the frequency, essentially overestimating it. As number of cells increase, the frequency converges downward toward the analytic value.
    d. Damping rate is a subtle thing to measure using PIC codes. Expected to be a non-monotonic graph that converges at moderate number of cells before getting noisier as number of cells increases.
3. How do the simulation time, noise level, frequency and damping rate vary with number of particles?
    a. Simulation time is expected to scale linearly with total particle count.
    b. Expect noise to decrease with the increase of number of particles :- noise is proportional to the inverse of the square root of the number of particles.

c. Variance in measured frequency should decrease as particle number increases as peaks are less noisy. Small shifts in frequency can occur at low number of particles due to discrete particle sampling and coarse deposition. This bias is expected to shrink as number of particles increases.
d. At low number of particles, measured damping has larger variance and sometimes bias which results in overestimated magnitude due to noise-driven fluctuations. As number of particles increases the decay window becomes clearer, fits are more stable and uncertainty decreases. However, convergence is often slower than that of frequency.

4. Plot each quantity (with appropriate error bars) as a function of number of cells/particles, respectively.
   a. See figures 2 and 3.
5. How does noise level vary with number of cells and number of particles?
   a. In a PIC code, noise decreases as $1/\sqrt{Np}$ and increases as $\sqrt{Nc}$ for fixed total particle number, because the dominant source of noise is finite macro-particle sampling. Therefore noise scales as noise $\propto \sqrt{(Nc/Np)}$. Fitted exponents are in good agreement with the theoretical +/-½ expectation.
6. Propose an equation to relate number of cells and particles to noise level
   a. The dominant contribution to PIC noise is statistical shot-noise from finite macro-particle sampling, which scales like (1/\sqrt{N_{p,\text{cell}}}), so increasing particles per cell suppresses noise efficiently.
   b. Because (N_{p,\text{cell}} = N_p/N_c), the predicted scaling is (\text{noise} \propto \sqrt{N_c/N_p}); simulations confirmed this trend, with fitted exponents close to the theoretical (+0.5) (cells) and (-0.5) (particles).
   c. A small constant offset in the fit indicates a numerical noise floor independent of particle statistics, likely tied to grid discretisation and floating-point rounding.
7. How does the computation time vary with number of cells and particles? Does it agree with what you would expect?
8. How do your results vary with number of cells and number of particles? Does your result converge?

---

9. 27/11/25 5pm (Session 2-3)

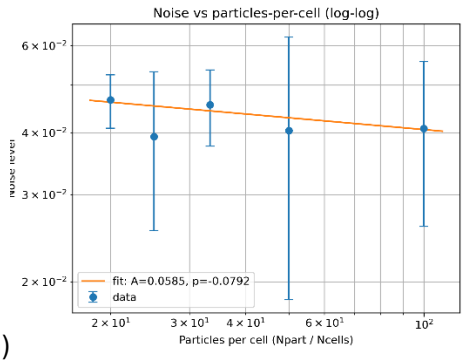
Results:

Fit (noise ≈ A * (Npart/Ncell)^p): A = 0.0585 ± 0.04198, p = -0.0792 ± 0.2168

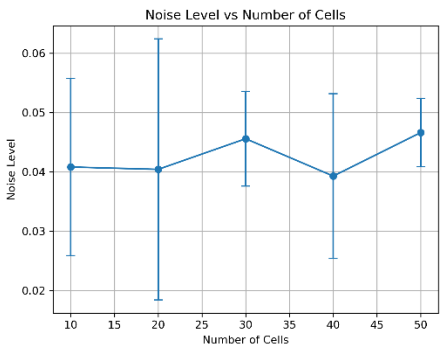
Fit (noise ≈ A * (Npart/Ncell)^p): A = 0.1981 ± 0.05862, p = -0.3945 ± 0.08278

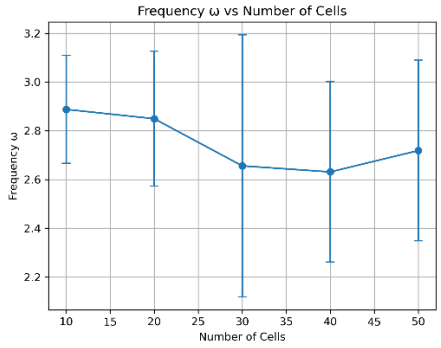
Cell Sweeps:

a)

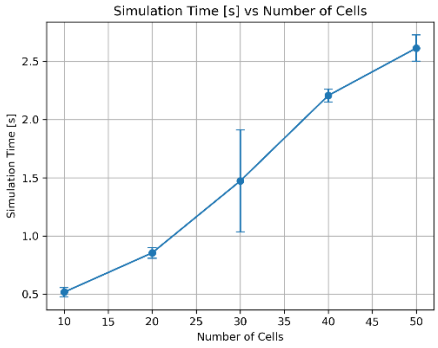


b)

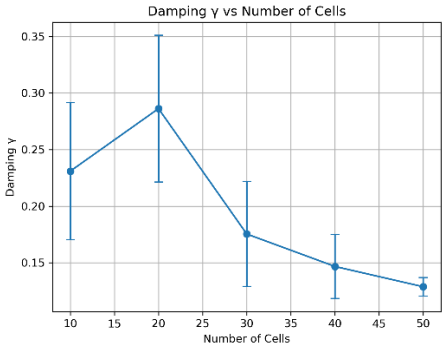




c)



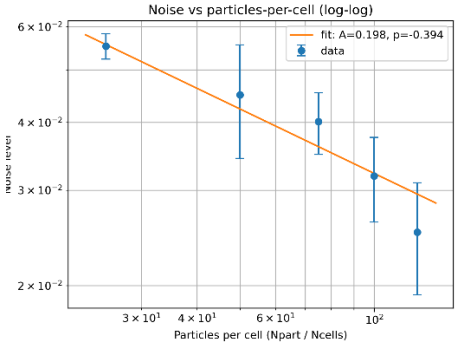
d)



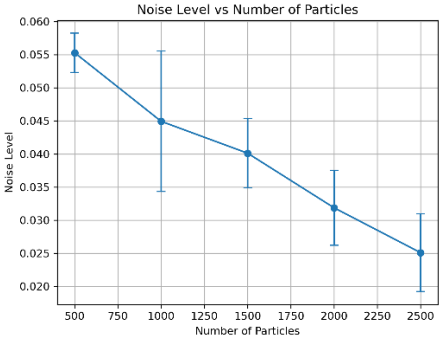
e)

Fig.2 a) Log-log comparison of noise vs particles per cell. The fit shows the inverse relationship between noise and number of particles, as expected. However, simulating with finer spatio-temporal resolution would most likely improve the visualisation/representation of this relationship. b) Noise is seen to increase for fixed number of particles and increasing number of cells, confirming the expected proportional relationship. c) Plot of langmuir mode frequency, showing the slight decrease in measured frequency before it converges as number of cells increases. This is expected behaviour, especially as diminishing returns after moderate resolution. d) Plot of simulation run time vs number of cells showing monotonic, almost linear relationship, as expected. e) Graph of damping vs number of cells showing expected relationship of biased damping due to numerical dispersion and coarse grid. Medium number cells is deemed the best estimate while if number cells was to increase further, noise domination is expected, resulting in fluctuations.

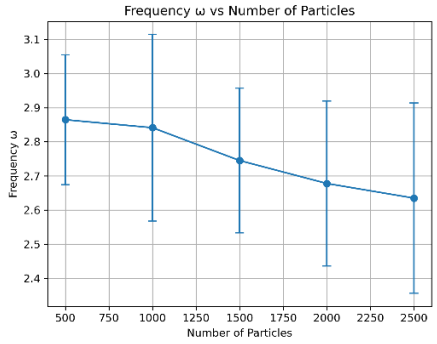
Particle Sweep:



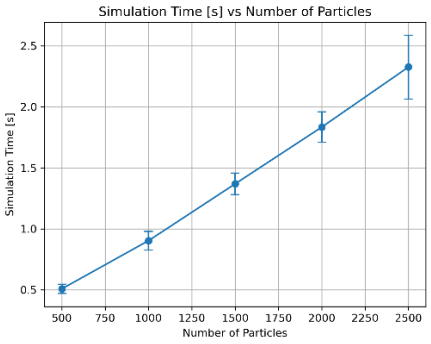
a)



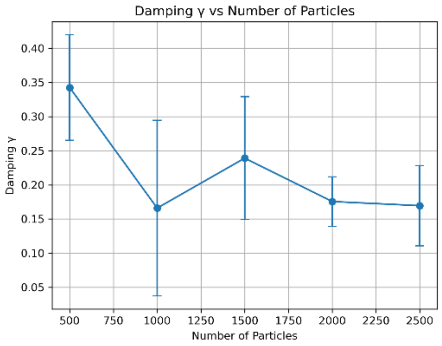
b)



c)



d)



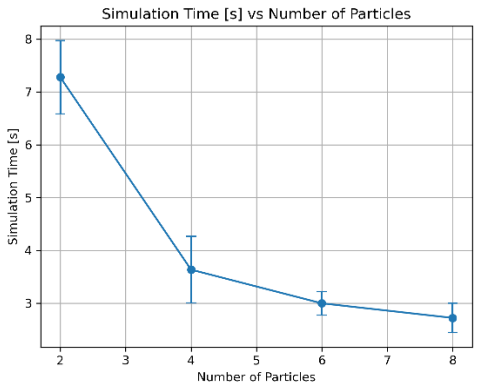
e)

Fig.3 a) Log-log plot of noise vs particles per cell showing approximate linear fit of the data, showcasing the inverse relationship between both variables. b) Plot of noise level vs number of particles confirms the inverse relationship between the two variables, with number of particles reducing the noise in the pic simulation and improving measurement accuracy. c) Frequency is seen to decrease as number of particles increases, showing the overestimation of frequency with higher particles and its convergence as the number of particles increases. d) Simulation time increases monotonically with number of particles, as shown in this plot, due to the increased complexity of this adds to the simulation. e)This plot illustrates the expected fluctuation of damping with lower values of simulation particles, with the convergence of damping values as the number of particles increases.

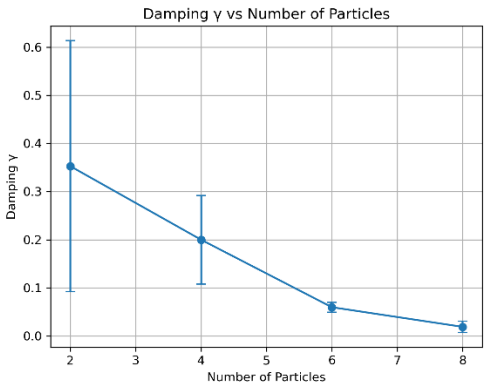
Physics Parameters

Fit (noise ≈ A * (Npart/Ncell)^p): A = 0.392 ± 3.9e+07, p = -0.6598 ± 2.543e+07

Length sweep:

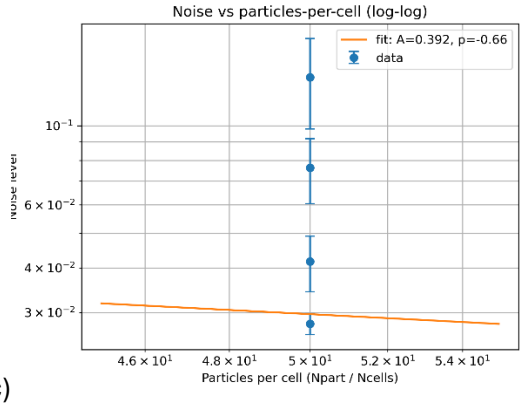


a)

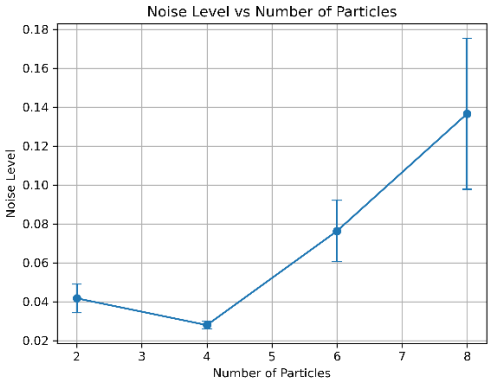


b)

c)



d)



e)

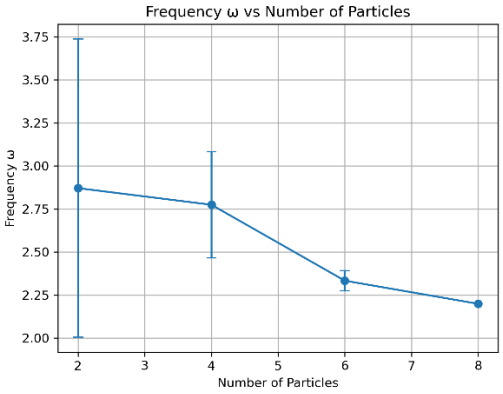


Fig.4 INCORRECT TITLES! DATA IS WRT. BOX LENGTH NOT PARTICLES PER CELL!

a) Plot of simulation time vs box length shows that while keeping number of cells and particles constant, runtime is reduced as there is less complexity over the simulation -> synonymous to coarser grid resolution. b) Damping depends on the wave number, k, excited in the box. Varying box length changes the accessible k-values. It is an inversely proportional relationship as illustrated by this plot. Variance is seen to decrease as box length increases, hinting at less dispersive effects. c) no idea what's going on here just yet... d) noise scales with particles per cell. Increasing box length while keeping number of particles constant reduces the particles per cell, therefore noise increases. This is illustrated in this plot. e) Similar to the situation in b, varying box length changes the accessible k-values. It is an inversely proportional relationship as illustrated by this plot. Variance is seen to decrease as box length increases, hinting at less dispersive effects and better approximation to continuum.

How do these quantities vary with box length? -> see caption of fig.4

04/12/25 12pm (Session 3-4)

---

1. Make the changes indicated above so that you can run the "default" two stream instability case and plot the first harmonic amplitude vs time. You should now see that the amplitude now increases in time until some point at which it saturates (i.e. stops growing).

   Done. See result in fig.5

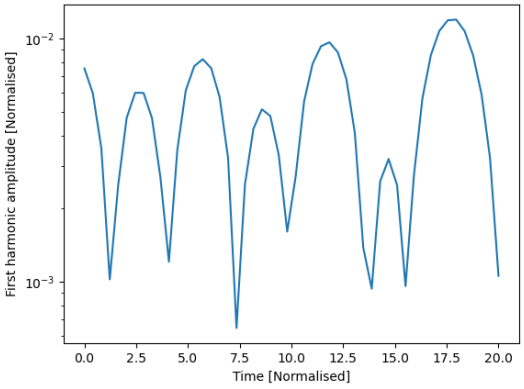


Fig.5 Resulting plot of the first harmonic amplitudes vs normalised time from the two stream instability simulation.

2. Restore the animation of the simulation (e.g. add p back into the diagnostics_to_run list) so you can see what is going on in the two stream case. What does the velocity distribution look like at the start of the simulation and once the amplitude is saturated? Can you comment on why the amplitude stops growing?

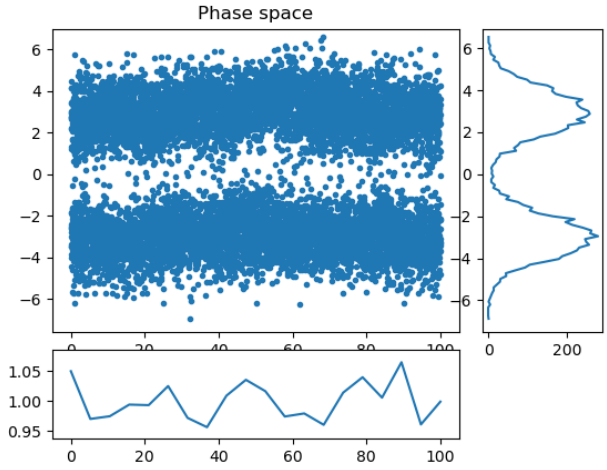


Fig.6 Phase Space animation close to beginnings of the simulation.

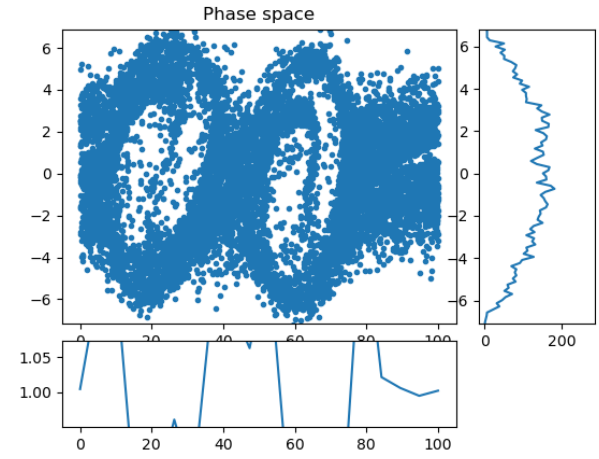


Fig.7Phase Space animation once amplitude is saturated.

Initially, the velocity distribution is seen to be two distinct beams centred about 0. The phase space plot shows two horizontal bands, as it is a cold plasma approximation (fig.6). Post-saturation, the distribution near the resonant velocity is seen to flatten (fig.7) signifying a trapped population, i.e. the flat region near the wave phase velocity. This manifests as two island vortices in the phase-space plot, which shows the trapped particle orbits. Overall, the distribution is hotter and smoother, as shown by the two separate beams being partially mixed and broadened. The amplitude stops growing due to nonlinear effects. The particle trapping and flattening of the velocity distribution at the resonant velocity reduces the free-energy that drive the two stream

instability. Exponential growth stops when the trapping frequency becomes comparable to the linear growth rate and the mode saturates. Mode-mode coupling and energy transfer into particle thermal energy and higher harmonics also contribute to this.

Measuring Growth Rate

1. Write some code to measure the growth rate. There are several approaches to this; one is to write code to identify the start and end of the growth phase and then fit over this region, whilst another is to fit a function to the whole time period which can accommodate the shape of the amplitude vs time data (e.g. a tanh function). Discuss you approach -- why did you choose it and what might the limitations be.

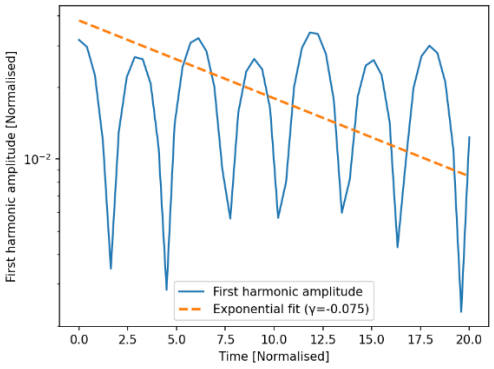


Fig.8 Growth rate γ = -0.0752 ± 0.0358; Fit interval: t = 5.31 → 20.00

Chosen method: Automatic identification of the linear growth phase and log-fit

I extracted the growth rate by first identifying the time interval in which the mode amplitude grows exponentially. This was done by computing the numerical derivative of ln A(t) and selecting the region where the derivative is roughly constant. Over this interval I fitted a simple exponential model A(t) = A_0 exp(\gamma t) using least-squares.

I chose this approach as the exponential growth is the defining feature of the linear instability phase. It also mitigates the bias that's imposed on the dataset fit due to noise-floor and nonlinear saturations that distort the signal. Lastly, the automatic selection of the linear region avoids hand-picking fit windows, which all in all makes the method more reproducible and robust.

A logistic/tanh-shaped model can be fitted to the entire amplitude curve, which includes the initial noise-floor and final saturation. However, this approach is more sensitive to starting guesses, assumes a particular nonlinear saturation shape, and can over-fit noise. Therefore, I chose the exponential-region method for reliability. However, it must be noted that if the instability is weak or noisy, the derivative may not exhibit a clean plateau, and an overly narrow interval may be chosen by the mask due to grid noise and finite-N fluctuations. The choice of threshold and smoothing must also be recognised to have an affect on the detected region.

2. Measure the growth rate from several repeat simulations of the growth rate. Report the average growth rate and a measure of the uncertainty.

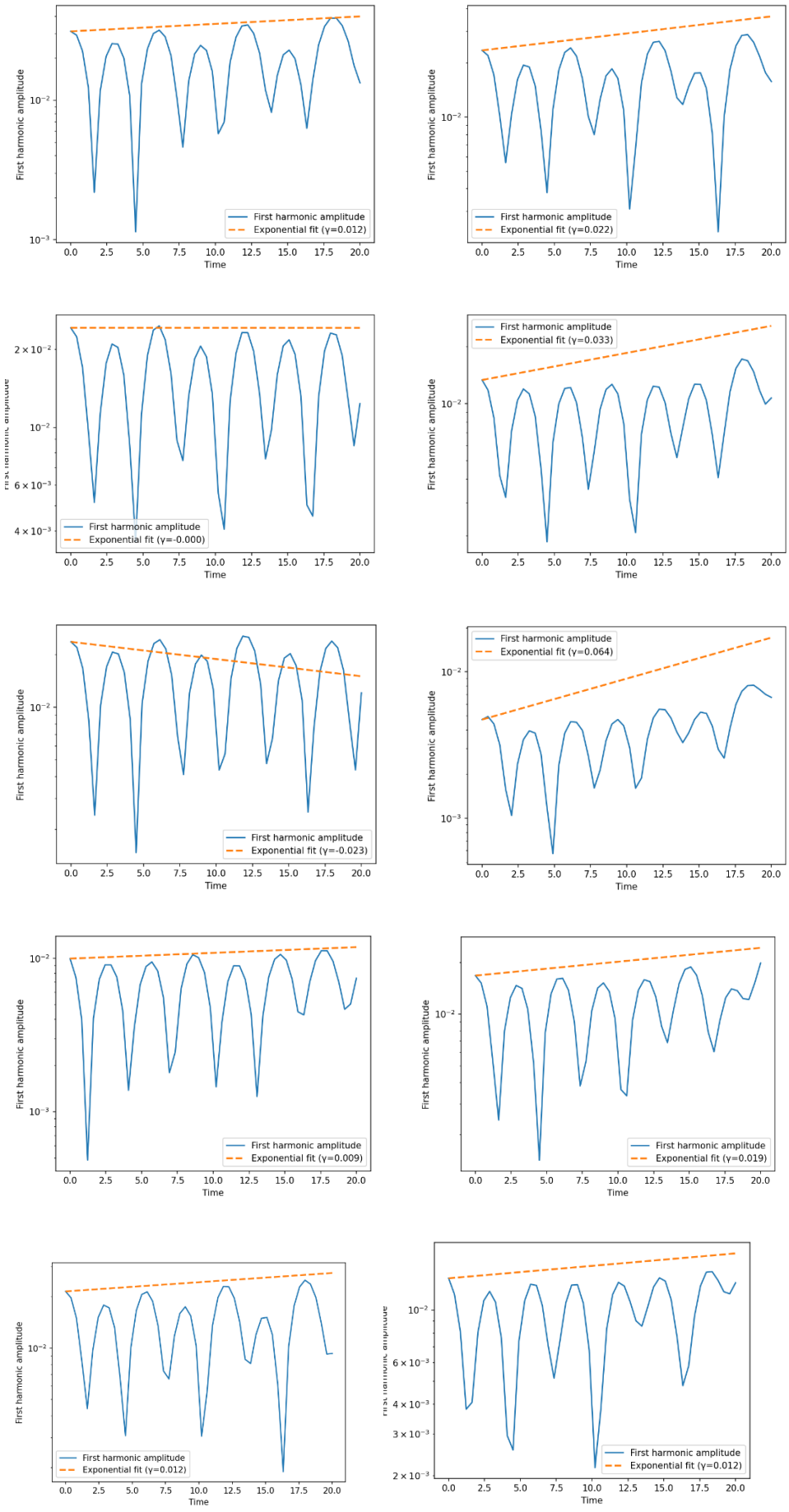


Fig.9 Repeated simulations to measure the growth rate. A lot of disagreement is seen between iterations. Future simulations need tweaking to become more accurate.

Growth rate statistics

Average γ = 0.0160

Std Dev = 0.0224 -> PRETTY BAD!!!

N = 10

04/12/25 3pm (Session 3-4)

We normalise density so the total uniform density is 1; therefore each cold beam has density $n_0 = 0.5$and the cold-beam dispersion predicts an upper instability threshold $kv_0 = \omega_{p0}\sqrt{2}$, which reduces to $v_{0,\mathrm{thresh}} = 1/k$. For $L = 100$(fundamental $k = 2\pi/L$) this gives $v_{0,\mathrm{thresh}} \approx 15.9155$. Sweeping $v_{\mathrm{beam}}$and measuring the growth rate (averaged over several runs to reduce finite-N scatter) shows an unstable band whose upper edge lies near the analytic cold-beam threshold; the lower edge is shifted above zero due to finite beam temperature. Reducing the beam thermal width by a factor of 10 moves the measured thresholds closer to the cold-beam prediction and increases the peak growth rates, confirming that finite temperature is the dominant source of disagreement.

First, density is normalised such that the total uniform density is 1. This means that each cold beam has density n_0=0.5 , with the cold-beam dispersion predicting an upper instability threshold kv_0 = \omega _ p0 sqrt(2). This reduces to v_0,threshold = 1/k. For a box length of 100, v_0,threshold = 15.9155.

The beam velocity is swept and the growth rate is measured and averaged over several runs to reduce the finite-N scatter. An unstable band becomes apparent where the upper edge lies near the analytical cold-beam threshold, while the lower-edge is shifted above zero because of the finite beam temperature.

Reducing the thermal width of the beam by a factor of 10 shifts the measured threshold closer to the cold-beam prediction and increases the peak growth rate. This confirms that the finite temperature is the dominant source of disagreement.

- Analytic (cold-beam) upper threshold:

  v_crit_theory = 15.91549

- Wide-beam PIC thresholds (σ = 1.0):

  Lower threshold ~ 1.7362357428206763

  Upper threshold ~ None

- Narrow-beam PIC thresholds (σ = 0.1):

  Lower threshold ~ 2.501006248586927

  Upper threshold ~ 15.00603749152156

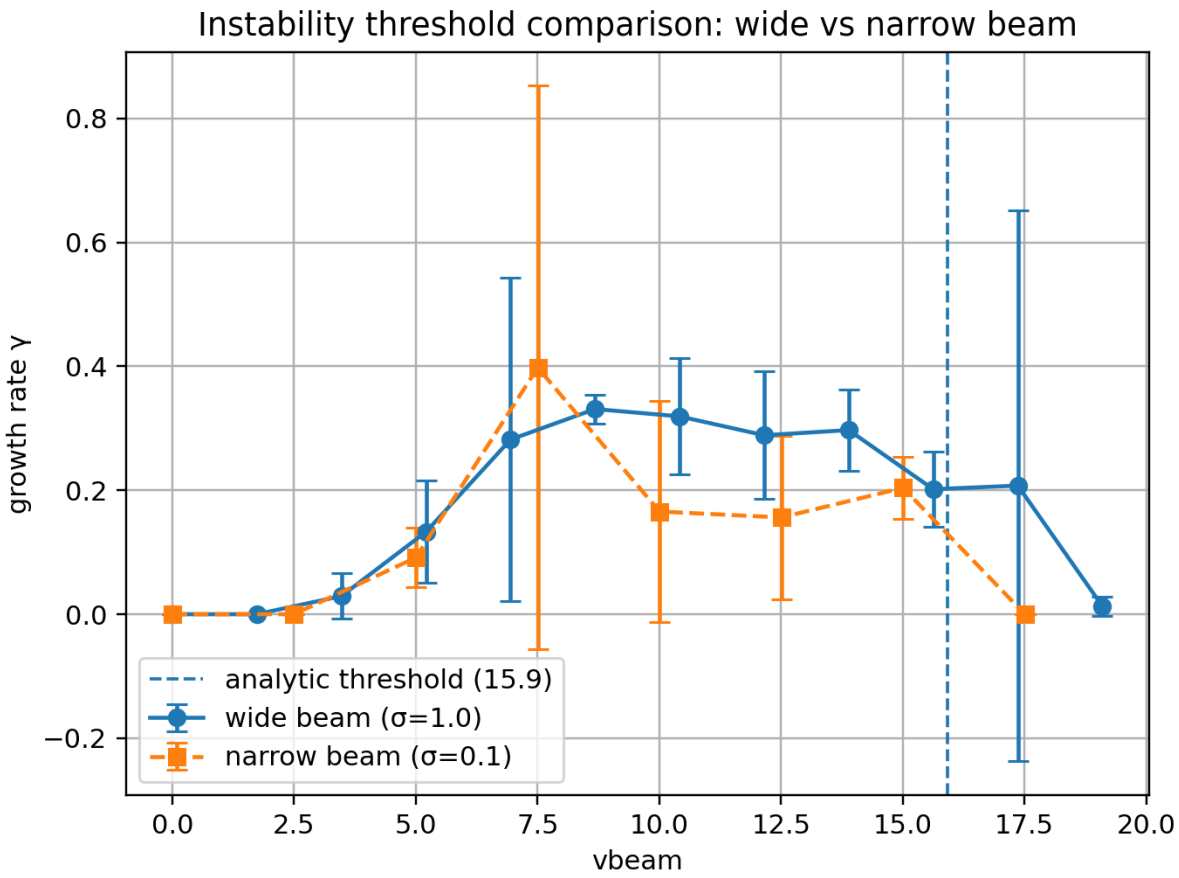


Fig.10 Plot of the instability threshold comparison between wide and narrow beams. Narrow beam proves favourable.

1. The wide-beam curve shows significant velocity spread, this means the instability band widens and the upper threshold sits above the analytic cold-beam prediction.

3. The narrow-beam produces thresholds closer to the analytic value, because the beam now approximates the cold-beam assumption.

4. The upper threshold tightens sharply as sigma tends to 0, showing convergence toward the analytic limit.

5. Remaining discrepancy probably comes from PIC noise, finite N, finite L, and finite-time growth-rate fitting.