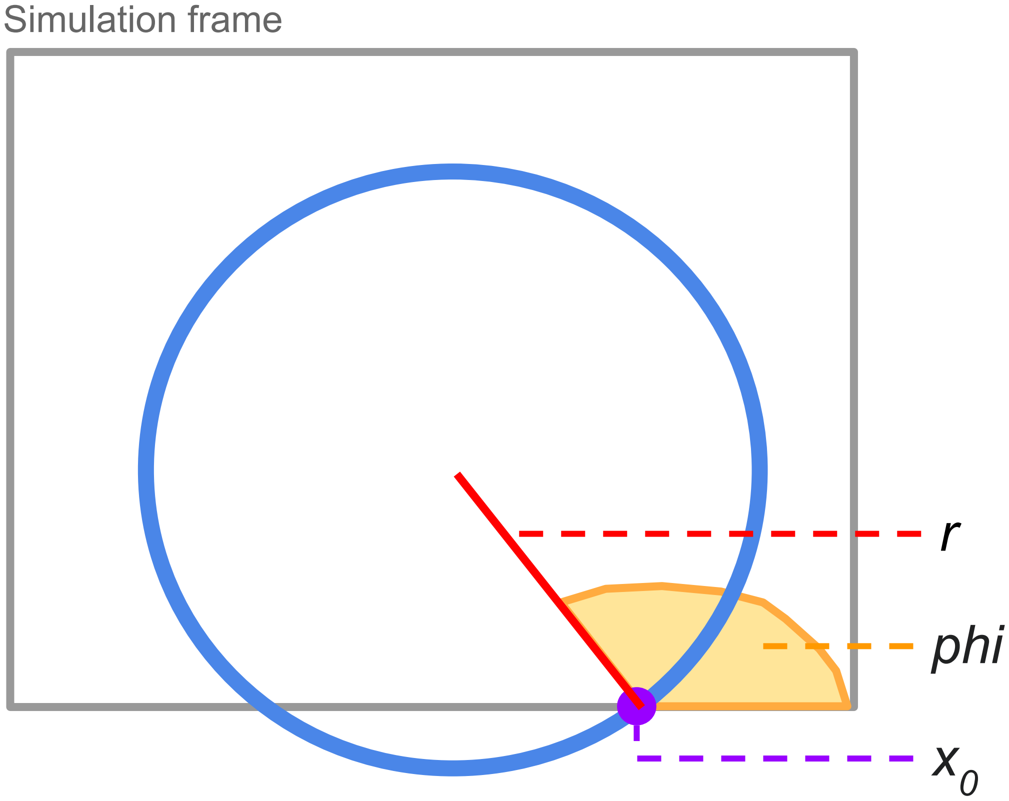
# Language Model Training with STT Toy Data Generator

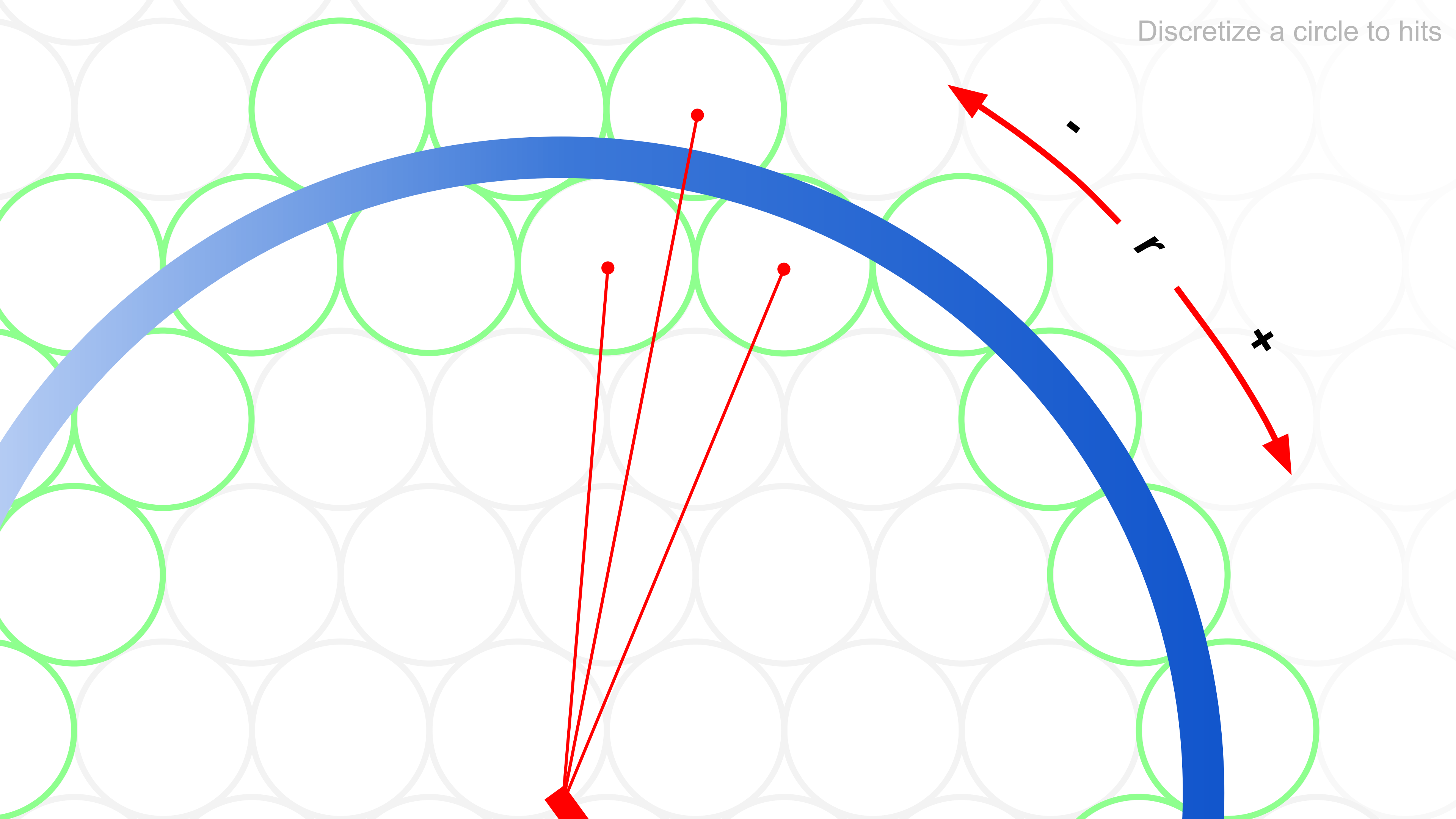
Jakapat Kannika

# Toy Data Generator for STT

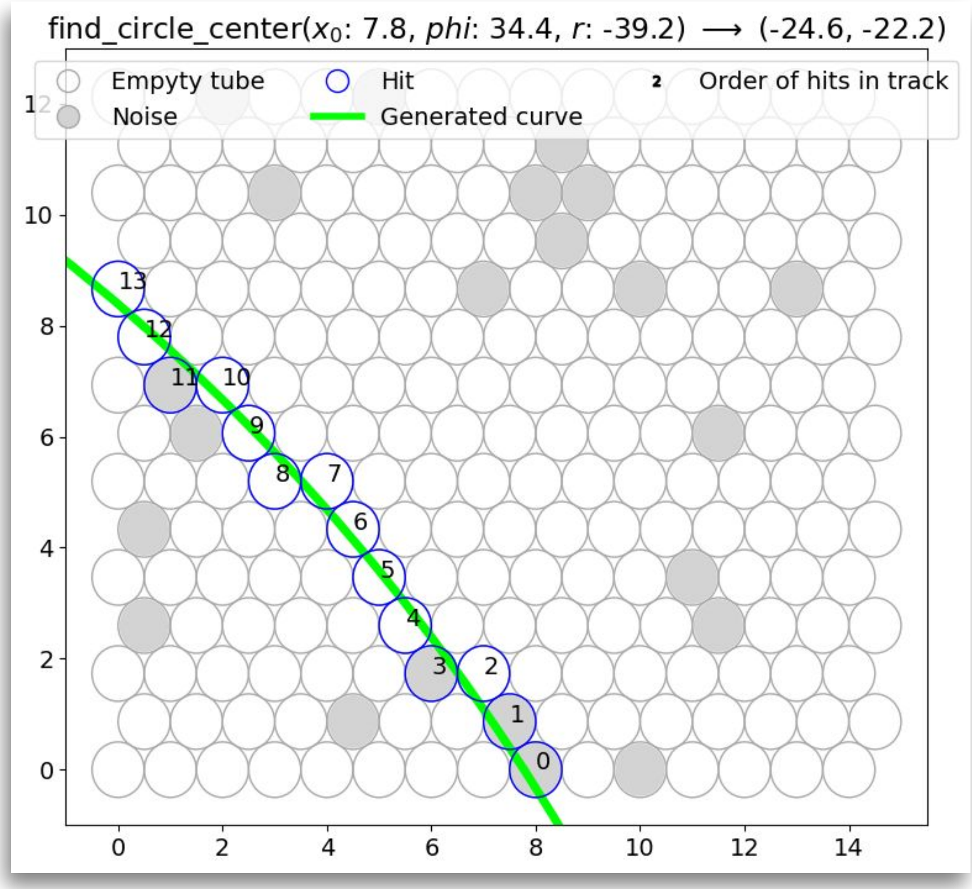
Generate track from a circle

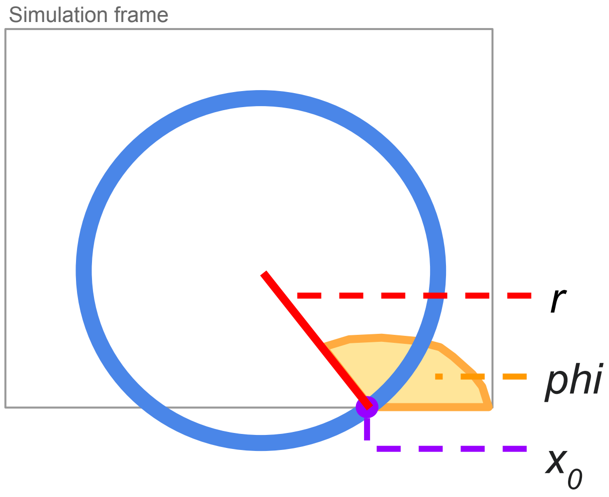


Discretize a circle to hits

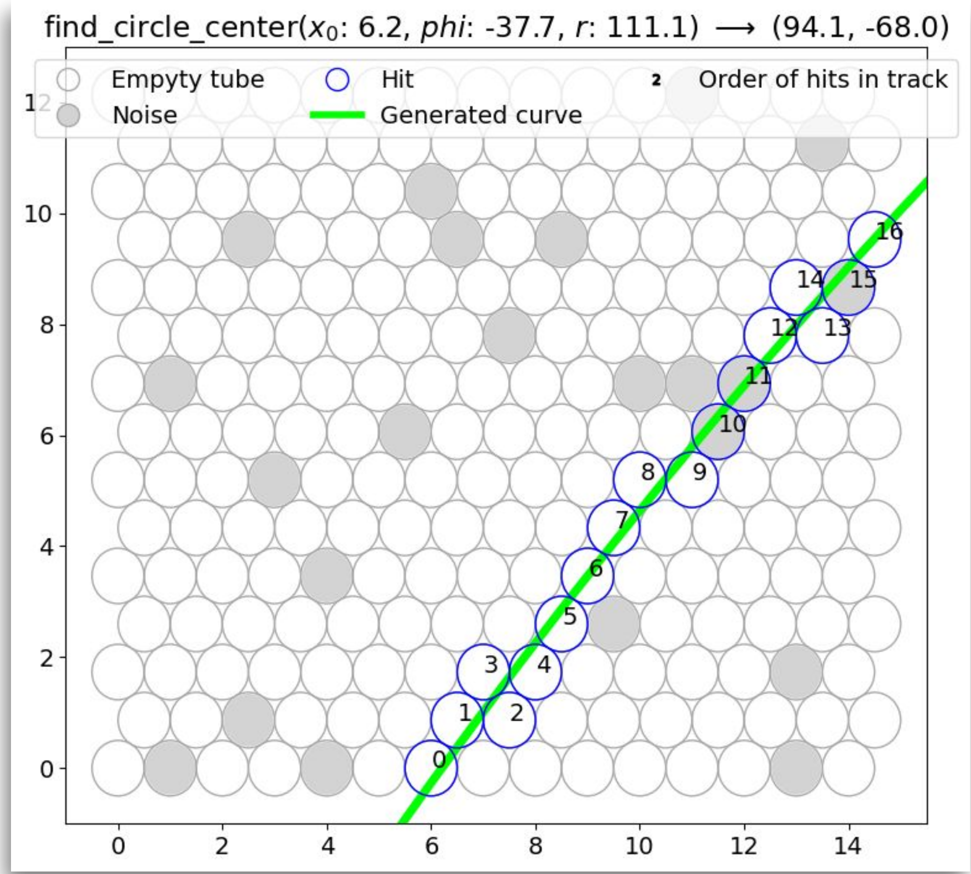


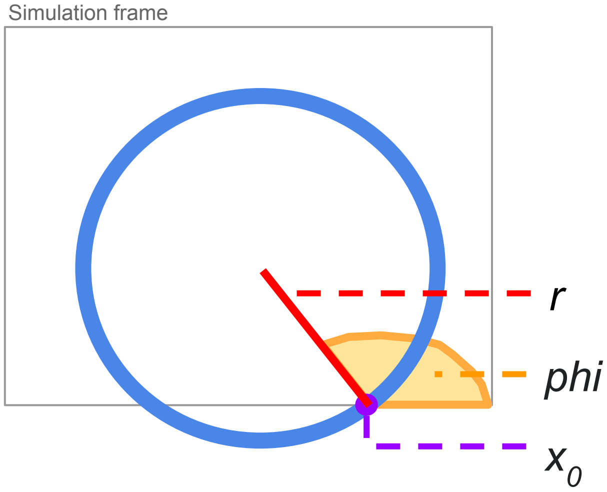
# Example of data generated by the toy data generator



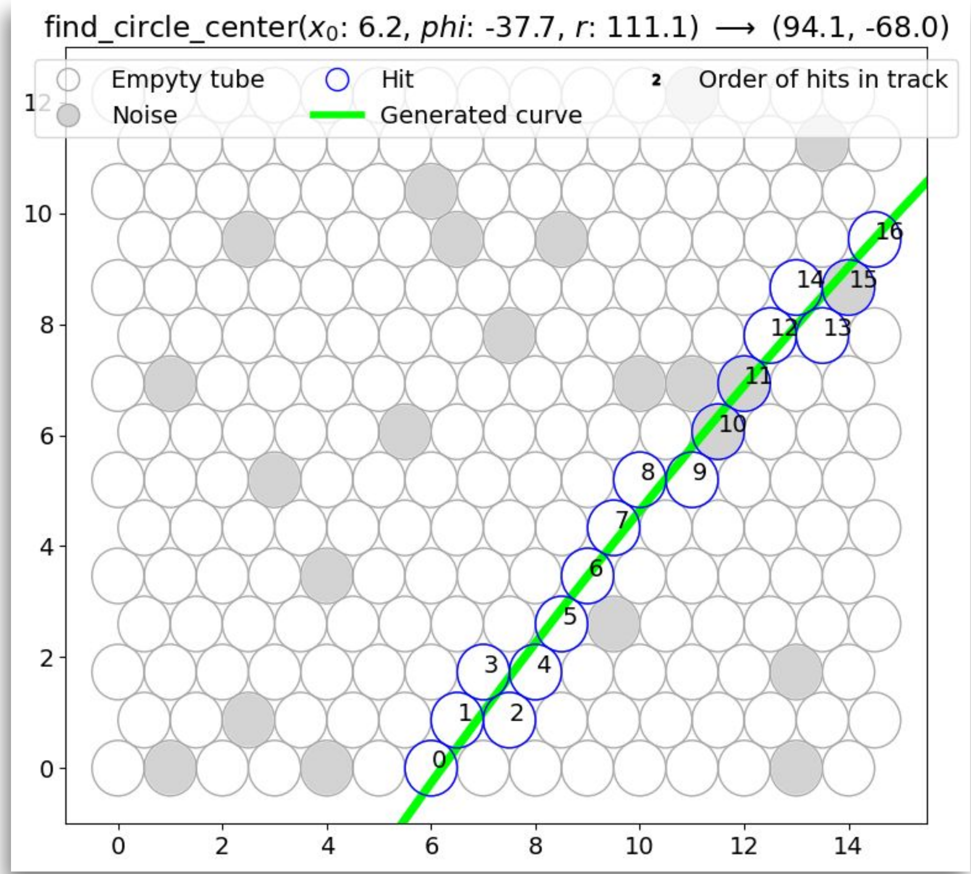


## Example of data generated by the toy data generator





## Example of data generated by the toy data generator



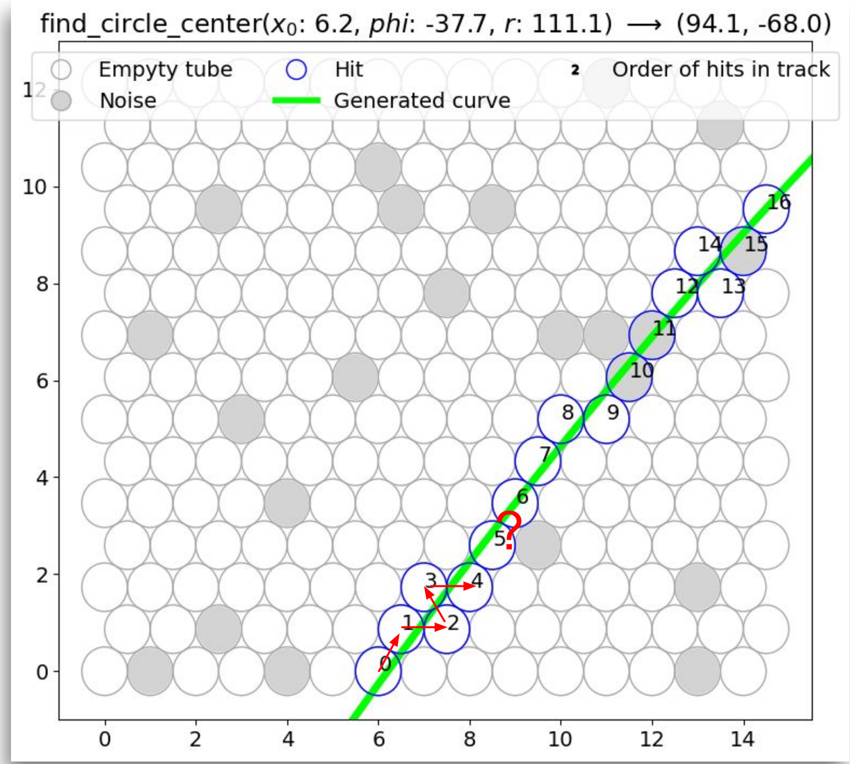
Positions: [
[6.0, 0.0], [6.5, 0.9], [7.5, 0.9], [7.0, 1.7], [8.0, 1.7], [8.5, 2.6], [9.0, 3.5], [9.5, 4.3], [10.0, 5.2], [11.0, 5.2], [11.5, 6.1], [12.0, 6.9], [12.5, 7.8], [13.5, 7.8], [13.0, 8.7], [14.0, 8.7], [14.5, 9.5]]

**Tracking features**

Moving directions: [
60, 0, 120, 0, 60, 60, 60, 60, 0, 60, 60, 60, 0, 120, 0, 60]

Neighbor patterns: [
[1, 41, 7, 56, 13, 41, 25, 9, 40, 5, 11, 13, 41, 7, 56, 13, 8]

# Language Model



A statistical language model is a probability distribution over sequences of words. Given such a sequence, say of length m, it assigns a probability $P(w_1, \ldots, w_m)$ to the whole sequence.*

…

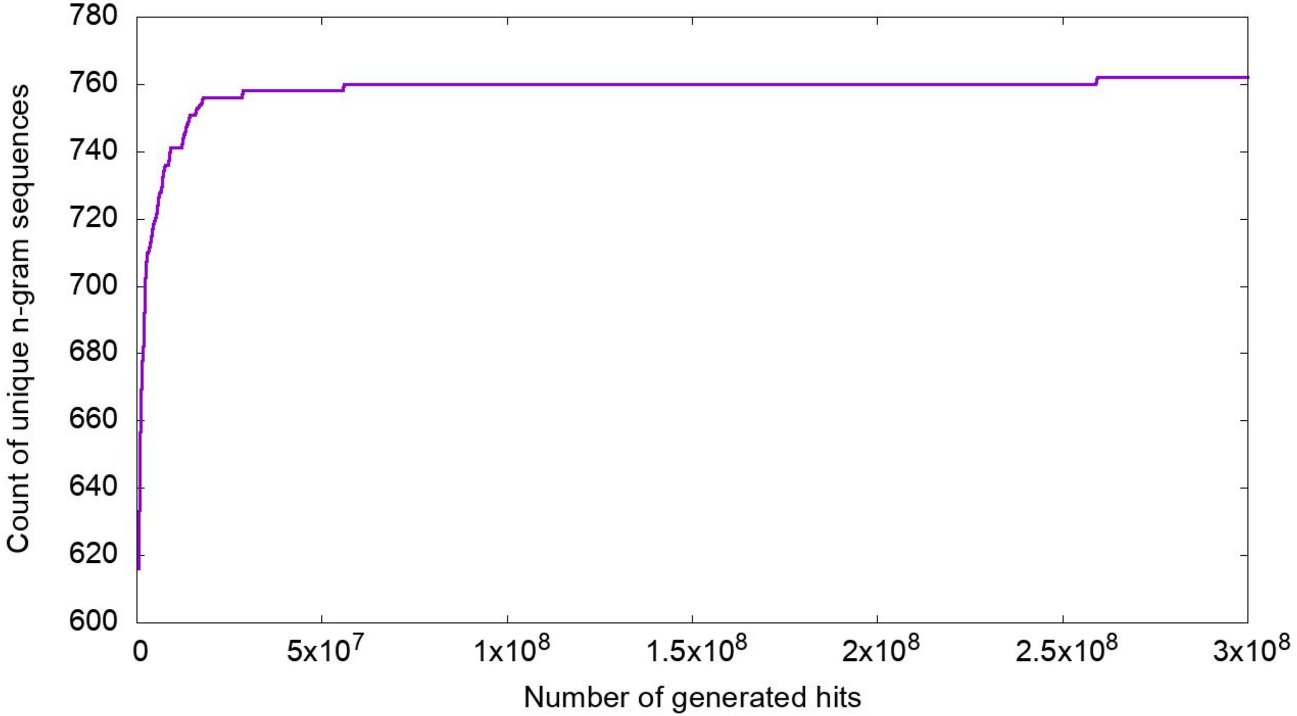
* https://en.wikipedia.org/wiki/Language_model

# Moving directions:

60, 0, 120, 0, 60, 60, 60, 60, 0, 60, 60, 60, 0, 120, 0, 60

| Pattern | Count | Prob. |
|---|---|---|
| 60, 0, 120, 0 | 2 | 1.00 |
| 0, 120, 0, 60 | 2 | 1.00 |
| 120, 0, 60, 60 | 1 | 1.00 |
| 0, 60, 60, 60 | 2 | 1.00 |
| 60, 60, 60, 60 | 1 | 0.33 |
| 60, 60, 60, 0 | 2 | 0.66 |
| 60, 60, 0, 60 | 1 | 0.50 |
| 60, 0, 60, 60 | 1 | 1.00 |
| 60, 60, 0, 120 | 1 | 0.50 |

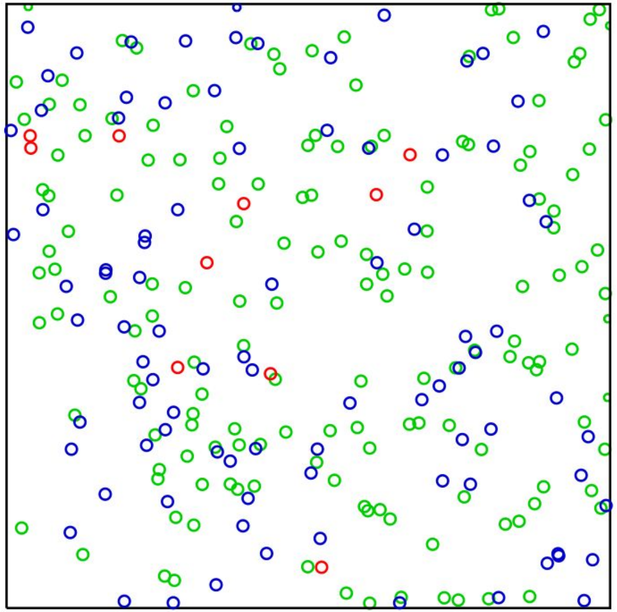
Training 4-gram model for moving directions


780
760
740
720
700
680
660
640
620
600
Count of unique n-gram sequences
0
5x10^7
1x10^8
1.5x10^8
2x10^8
2.5x10^8
3x10^8
Number of generated hits

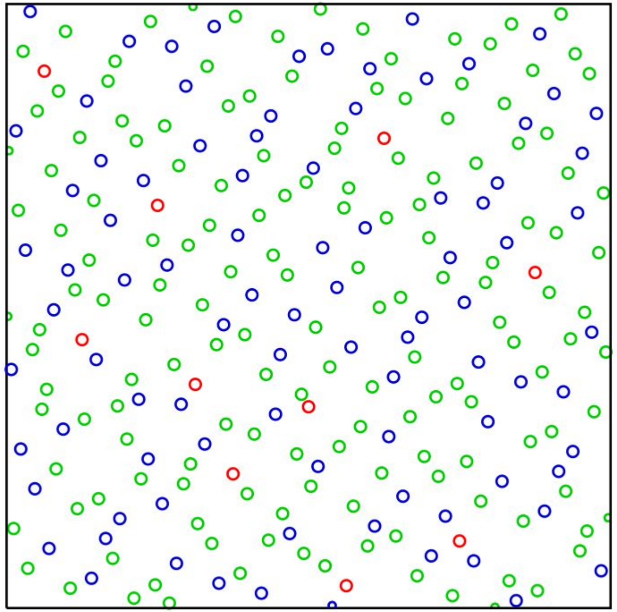
# Current training models

- Training feature: moving directions,
- Language models: 5-gram, 10-gram, 15-gram models,
- Sizes of simulation frames: 15 x 15, 20 x 20, 25 x 25 tubes
- Noise: 0 noise hit.

# Optimize training speed using halton sequence



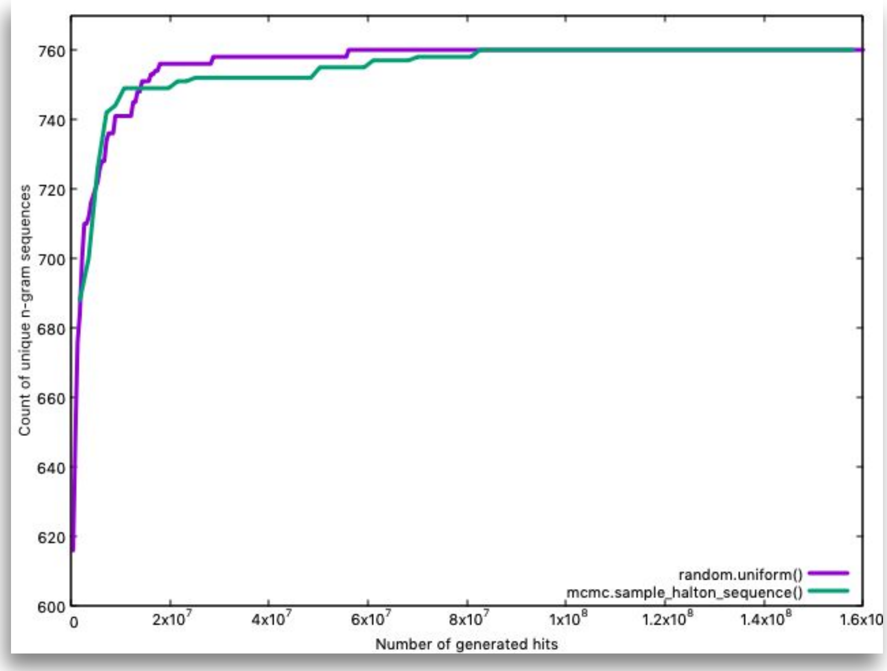
Pseudorandom



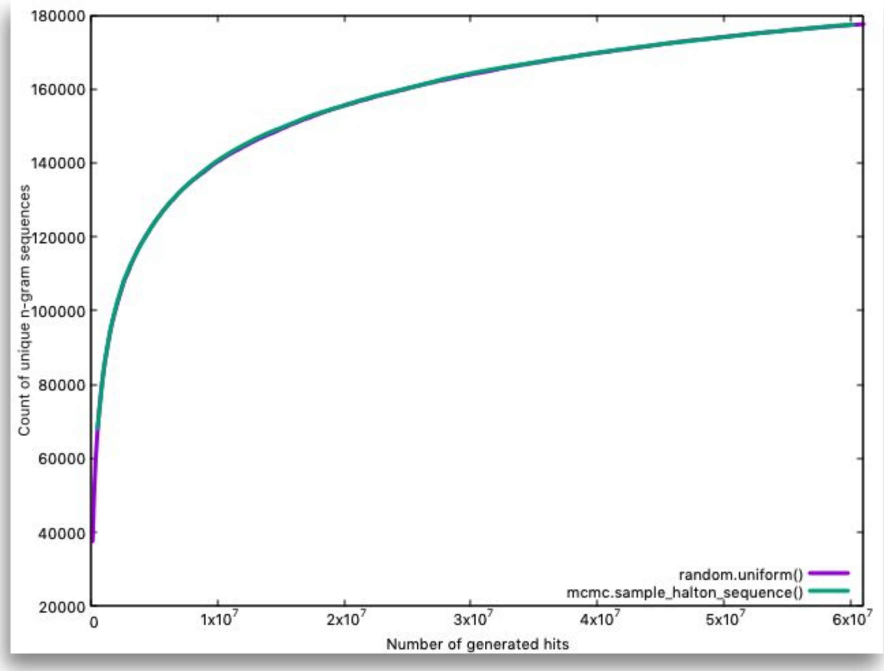
Halton sequence

src.: https://en.wikipedia.org/wiki/Halton_sequence

Speed of training of pseudorandom vs Halton sequence



5-gram model



10-gram model

# Check distributions of hits

Simulation frame:

- Width = 15 tubes,
- Height = 15 rows.

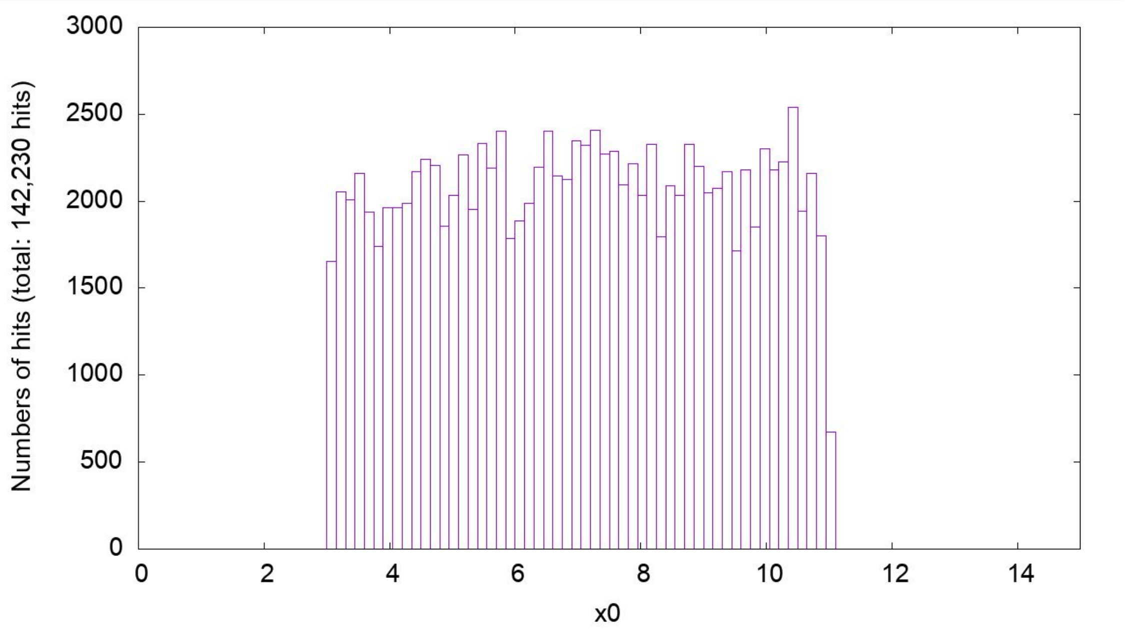
Training language model:

- 5-gram model for moving directions.

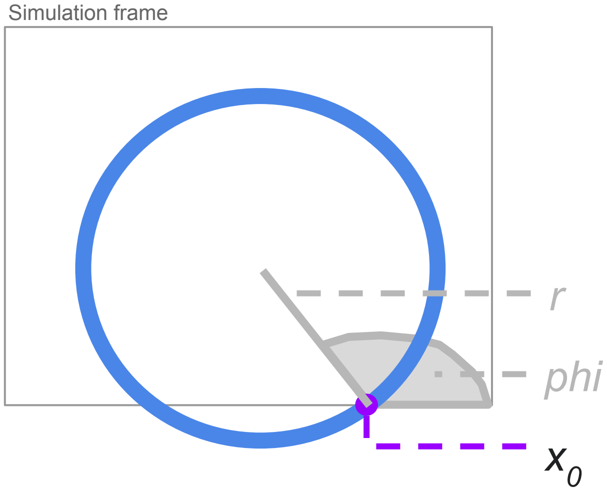
Number of generating data:

- 142,260 hits (10,000 tracks)
- 0 noise hit.

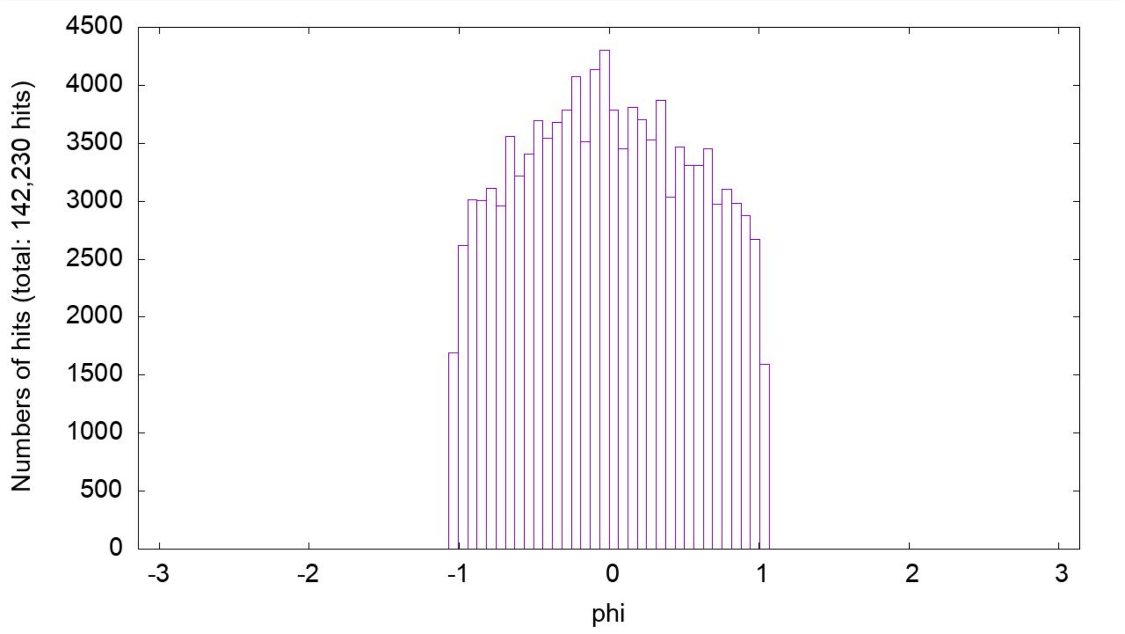
# Distributions of hits vs. parameters for generating track



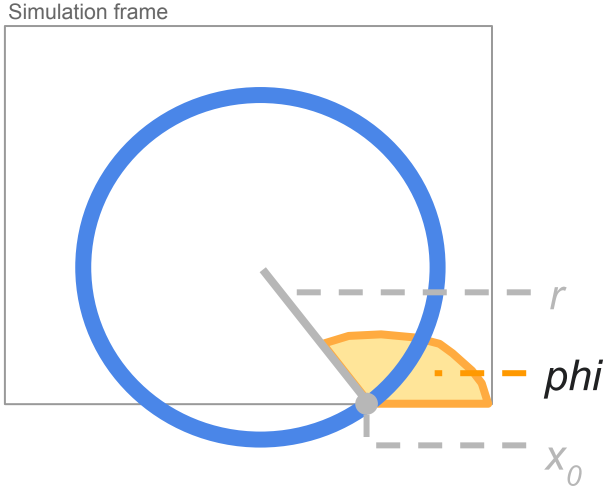
x0 = random.uniform(3, 11)



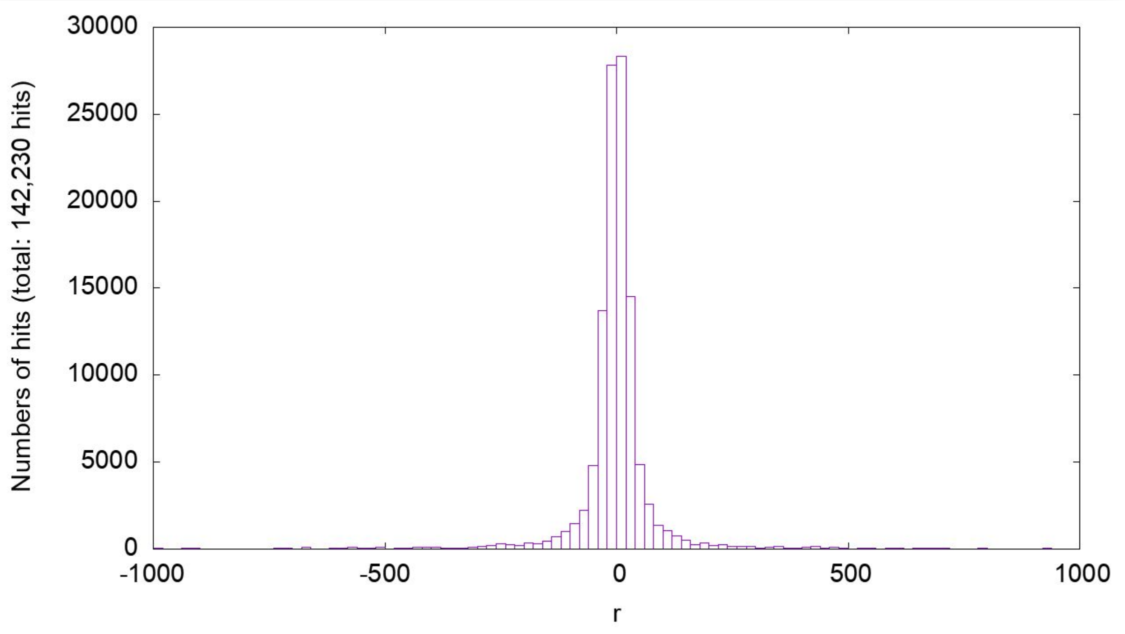
# Distributions of hits vs. parameters for generating track



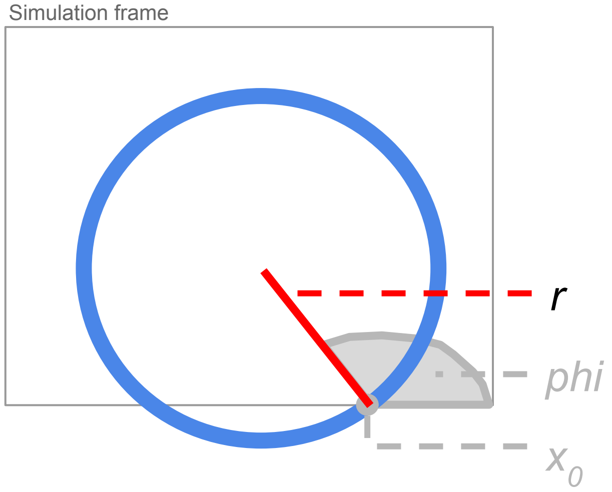
phi = random.uniform(-1 * math.pi / 3.0, math.pi / 3.0)

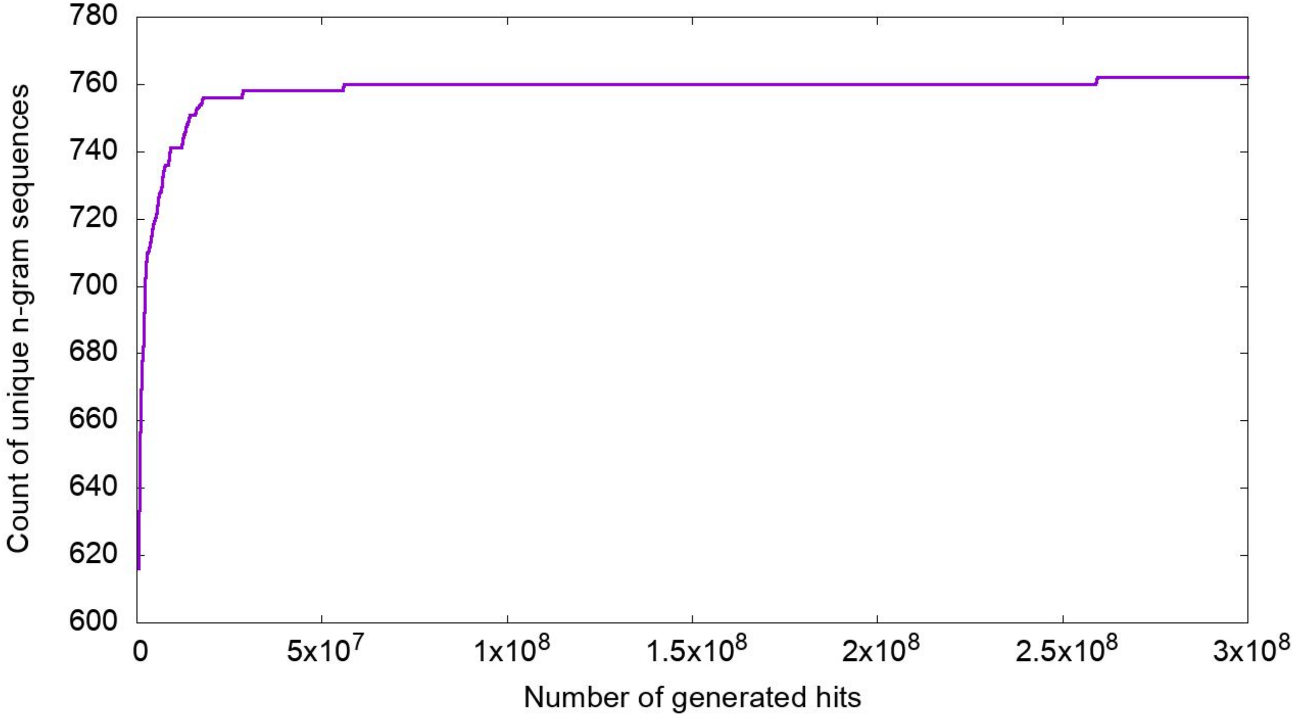


## Distributions of hits vs. parameters for generating track



```
a = random.uniform(0.001, 0.1)
r = random.choice([-1, 1]) * (1 / a)
```





780
760
740
720
700
680
660
640
620
600
Count of unique n-gram sequences
0
$5x10^7$
$1x10^8$
$1.5x10^8$
$2x10^8$
$2.5x10^8$
$3x10^8$
Number of generated hits

# Check distribution of new patterns

Simulation frame:

- Width = 15 tubes,
- Height = 15 rows.

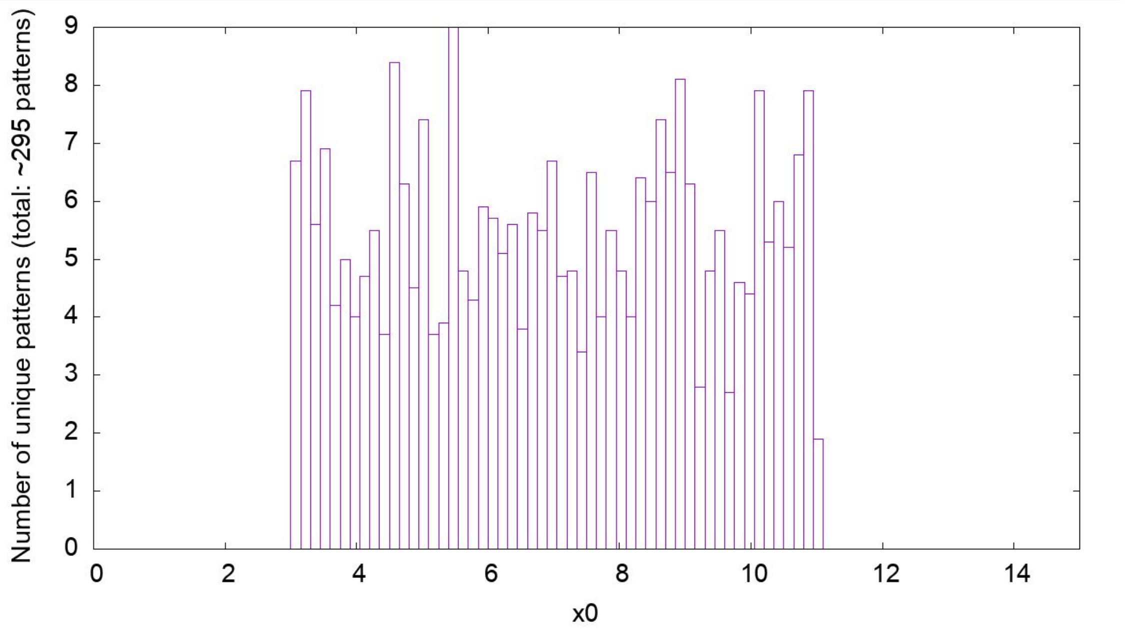
Training language model:

- 5-gram model for moving directions.

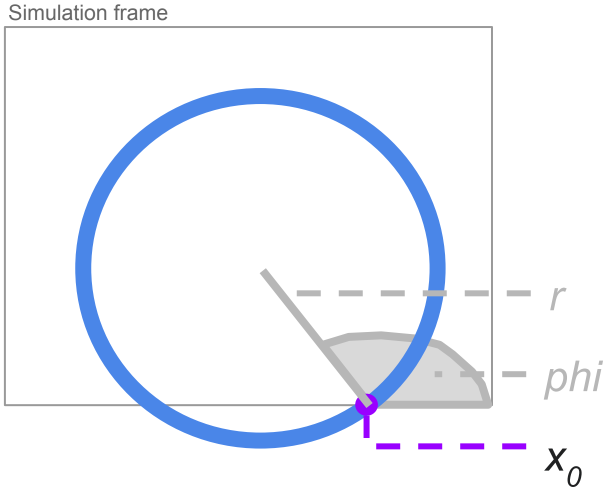
Number of generating data:

- ~141,982 hits (10,000 tracks)
- 0 noise hit.

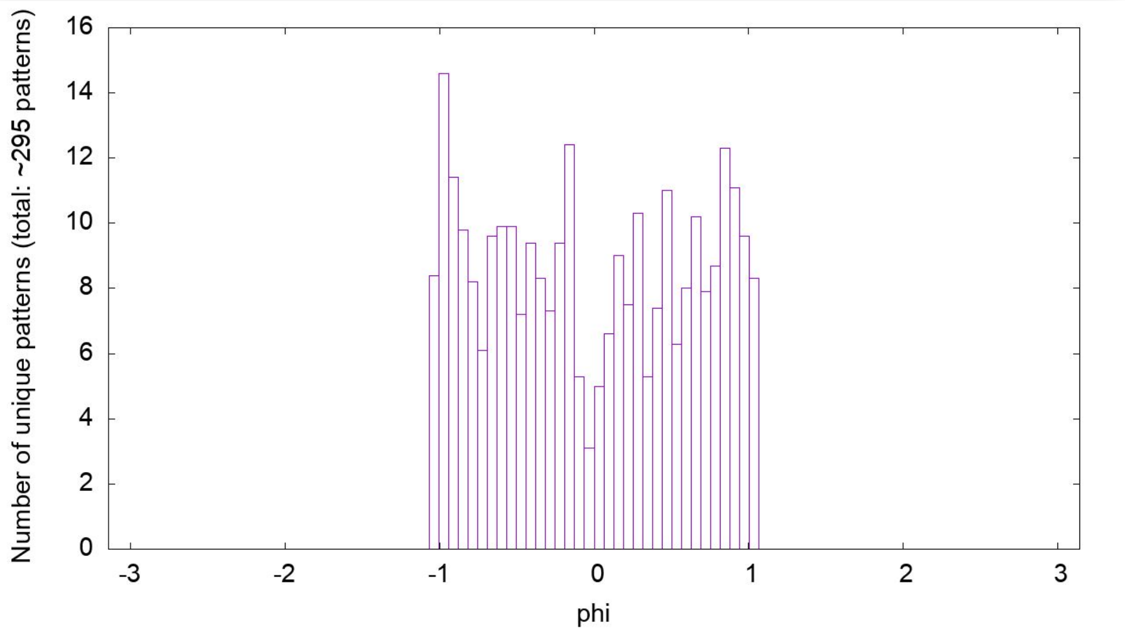
# Distributions of new patterns vs. parameters for generating track



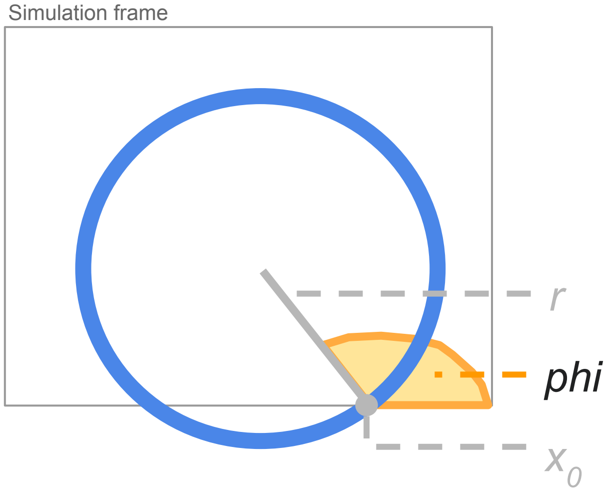
x0 = random.uniform(3, 11)



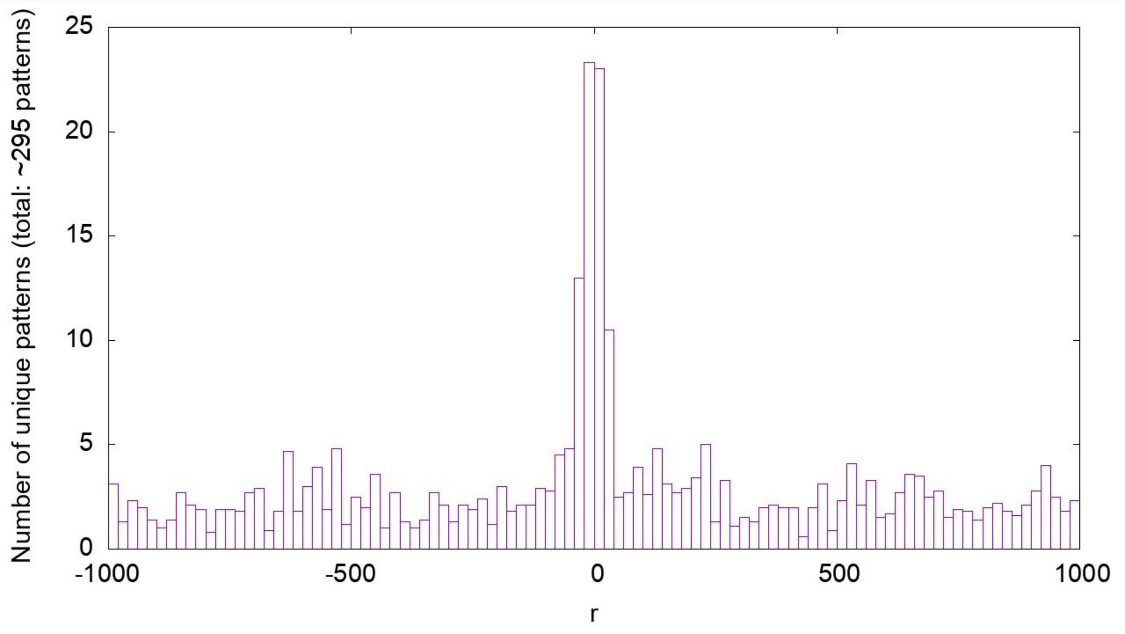
# Distributions of new patterns vs. parameters for generating track



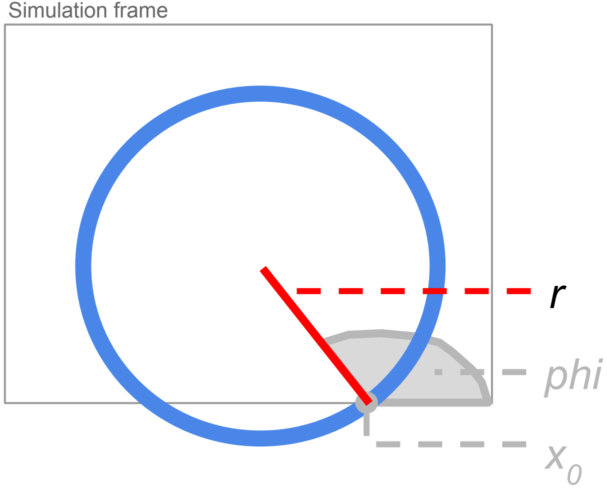
phi = random.uniform(-1 * math.pi / 3.0, math.pi / 3.0)



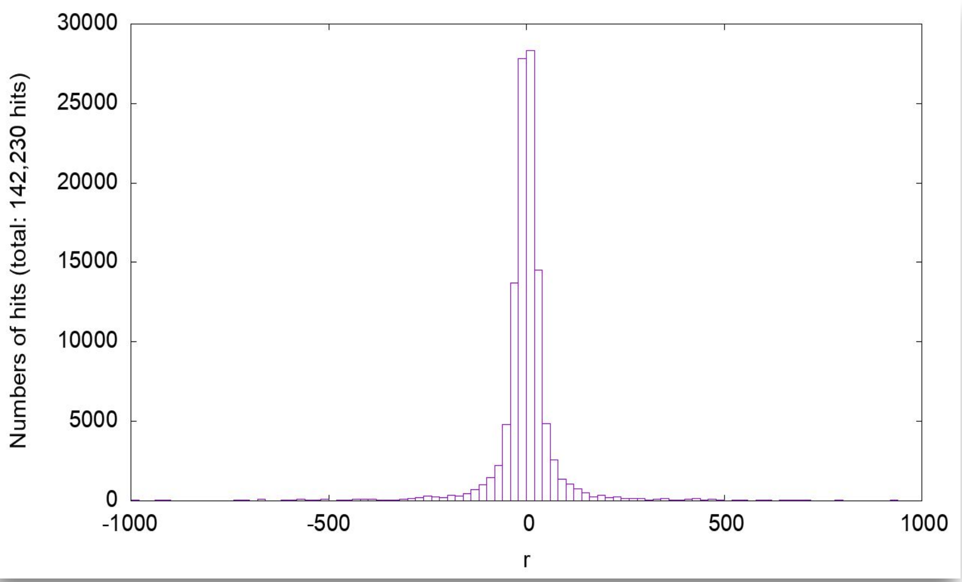
# Distributions of new patterns vs. parameters for generating track



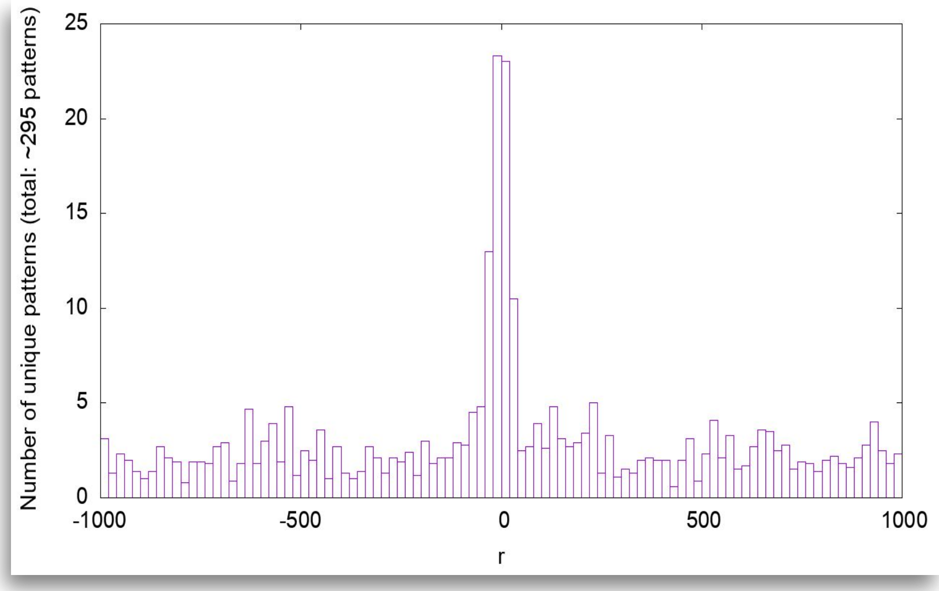
r = random.choice([-1, 1]) * random.uniform(10, 1000)



# Generated hits vs learning patterns



a = random.uniform(0.001, 0.1)
r = random.choice([-1, 1]) * (1 / a)



r = random.choice([-1, 1]) * random.uniform(10, 1000)

# Check for a bottleneck in the data generation

Simulation frame:

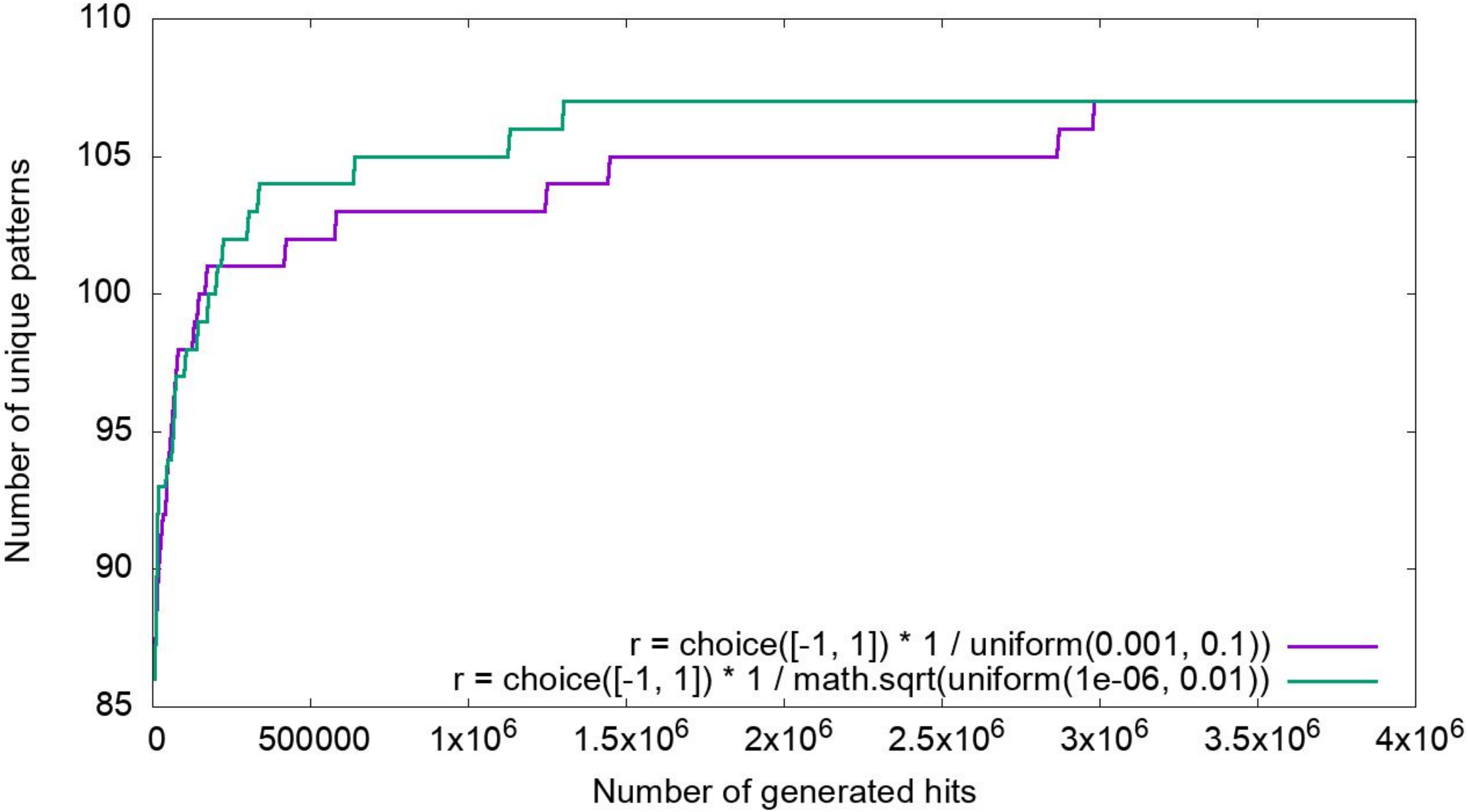
- Width = 15 tubes,
- Height = 15 rows.

Training language model:

- 3-gram model for moving directions.

Number of generating data:

- ~4,000,000 hits
- 0 noise hit.


110
105
100
95
90
85
Number of unique patterns
0
500000
1x10$^6$
1.5x10$^6$
2x10$^6$
2.5x10$^6$
3x10$^6$
3.5x10$^6$
4x10$^6$
Number of generated hits
r = choice([-1, 1]) * 1 / uniform(0.001, 0.1))
r = choice([-1, 1]) * 1 / math.sqrt(uniform(1e-06, 0.01))

# Summary and outlook

Summary:

- The new toy data generator can generate data with the geometry similar to the STT,
- The generator can produce consistent patterns that can be used in language model training,
- Feature extractors for moving directions and neighbor patterns are available for the new geometry,
- Slow in speed of training could be caused by a bottleneck in the data generation.

Outlook:

- Finish language model training for moving directions and neighbor patterns,
- Implement isochrone radius for the new data generator.