



Gemini

Arjun Ray

Undergraduate Research Assistant II
raya@frib.msu.edu

□ *The Joint DNP/JPS workshop "Fragment Separation at RIBF and FRIB and Rare Isotope Production Mechanisms"*



U.S. DEPARTMENT OF
ENERGY

Office of
Science



MICHIGAN STATE
UNIVERSITY



National
Science
Foundation

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Nuclear Physics and used resources of the Facility for Rare Isotope Beams (FRIB) Operations, which is a DOE Office of Science User Facility under Award Number DE-SC0023633, and by the US National Science Foundation under Grants No. PHY-20-12040 and 23-10078 "Windows on the Universe: Open Quantum Systems in Atomic Nuclei at FRIB".

Initial Optimization

– Gemini profiling to search for bottleneck points

– Gemini optimization at bottleneck points and more to result in Gemini being ~1.6x faster

A=195,J=50, Events= 3000	23.63	15.43	34.68%
Average speed increase: 38.41% faster, 1.6× faster			

BOTTLENECKS ADDRESSED AND CHANGES MADE				
No.		Location	Bottleneck	Change Made
1		Nucleus.cpp(CNucleus::getSumTl())	storeSub.push_back() caused heavy reallocations and copies	Reused a thread_local buffer, reserved capacity using (lMax - lMin + 1), and replaced push_back with emplace_back
2		Nucleus.cpp(CNucleus::getSumTl())	Repeated calls to getTl(L, ek*scale, temp) for same inputs	Added a small local cache to store computed getTl values and reused them
3		Nucleus.cpp (decay-product generation)	Heavy new/delete churn for CNucleus objects during decay	Introduced a recycle pool (acquire() / release()) to reuse CNucleus objects instead of allocating each time
4		Nucleus.cpp (decay-product generation)	Deep recursive decay traversal causing function call overhead	Replaced recursive decay traversal with an explicit stack-based loop
5		Project-wide	Extensive use of pow() in performance-critical code paths	Replaced pow() with equivalent direct arithmetic (e.g., x*x)
6		Nucleus.cpp (constructors & initialization)	Duplicated constructor logic and repeated setup code	Centralized all setup logic into initializeDefaults() and initialize(...)
7		Nucleus.cpp (decay-product handling)	Frequent temporary vector allocations across events	Reused decay-product vectors across events instead of recreating them
8		gm_mwExecFusion.cpp, gm_mwExecCompound.cpp	Throttled progress + cancel handling	Updated progress bar only every 10 iterations to reduce UI overhead

Angular Distribution Table

IMF emission

☒ use in calculations

☐ enhanced IMF emission

☐ Angular Distribution

Checkbox to display tables and plots in a new window

7. Energy and angular distribution of residual nucleus Z = 11 and N = 12 (²³Na)

Residual velocity/c Vz = 8.29e-02 (sig = 4.77e-03) rms Vxy = 9.30e-03

Energy Range	Angular range (deg)																	
(MeV)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
54.0 - 60.0				2														
60.0 - 66.0		1			1	1	1				1		1					
66.0 - 72.0		2	4	3	3	1	1	1					1				1	
72.0 - 78.0			1		1	2		2	1									
78.0 - 84.0			2	2	1	3	1		3	1	1	1						
84.0 - 90.0	1	2	3	1						1	1							
Above 174																		
Total	1	5	10	8	6	7	3	3	4	2	3	1	2				1	
dSig/dOmeq	1.1e+03	1.8e+03	2.2e+03	1.2e+03	7.2e+02	6.9e+02	2.5e+02	2.2e+02	2.5e+02	1.1e+02	1.5e+02	47	87	0.00	0.00	0.00	33	0.00
0 - 0																		
0 - 90	1	5	10	8	6	7	3	3	4	2	3	1	2				1	
90 - 180																		
Above 180																		

[Plot E vs Theta](#) [Plot N vs Theta](#) [Plot N vs E](#) [Plot dCS/dTheta vs Theta](#)

– Implemented PACE style angular distribution tables for Gemini

4 types of plots for each table



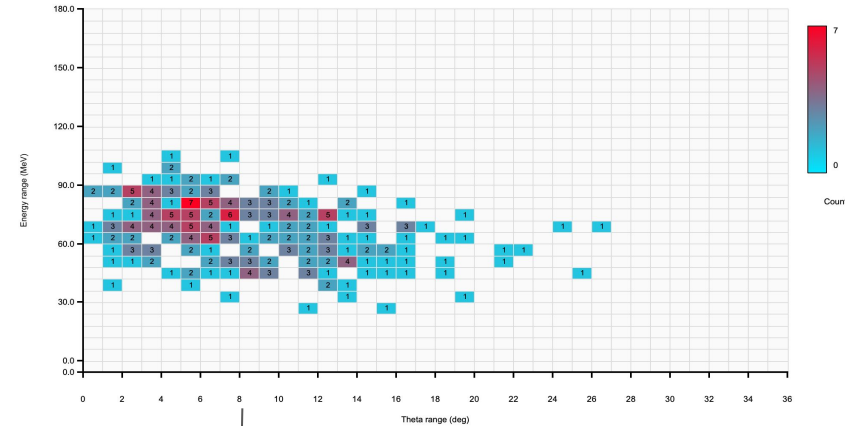
Facility for Rare Isotope Beams
U.S. Department of Energy Office of Science | Michigan State University
640 South Shaw Lane • East Lansing, MI 48824, USA
frb.msu.edu

Plots

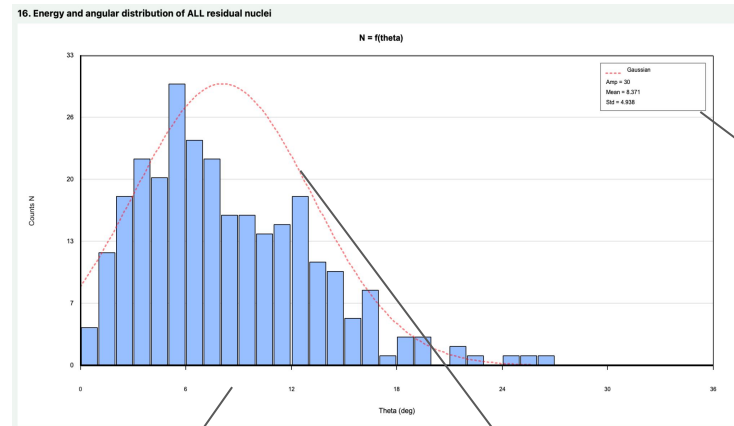
[Plot All: E vs Theta](#) [Plot All: N vs Theta](#) [Plot All: N vs E](#) [Plot All: dCS/dTheta vs Theta](#)

“Plot All” option for each type of plot to display plots for all tables

*** Spin alignment perpendicular to recoil axis - standard compound nucleus angular distribution



2D E vs Theta Plot example



1D plot example

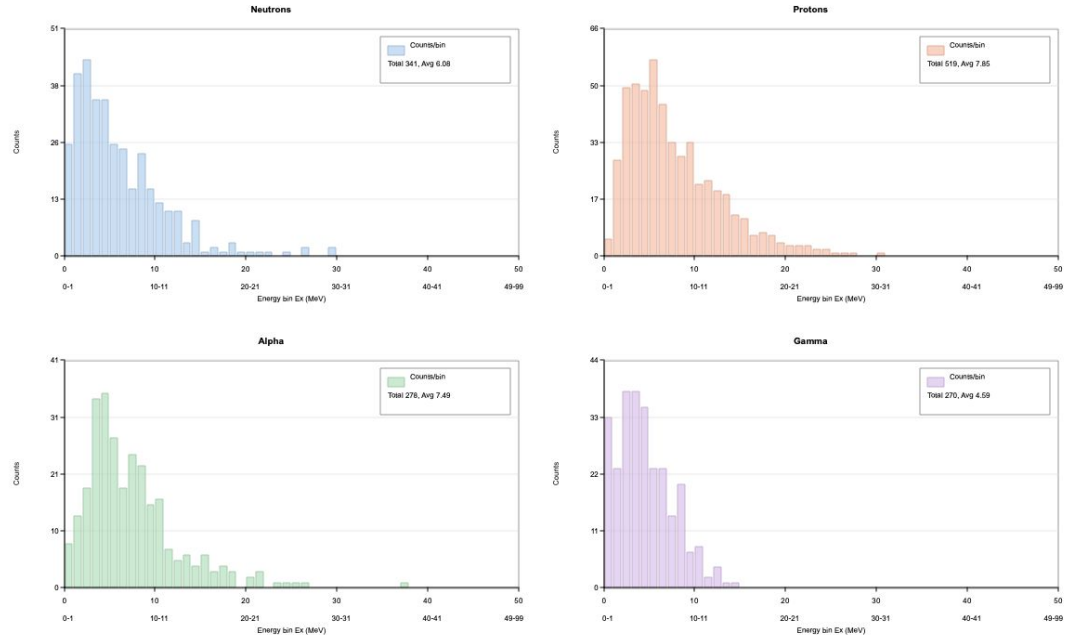
Gaussian distribution for all 1D plots with N(count)

C.M Spectra

Plot C.M. spectra

Ex (MeV)	Neut	Prot	Alpha	Gamma
0 - 1	25	5	8	33
1 - 2	41	28	13	23
2 - 3	44	49	18	38
3 - 4	35	50	34	38
4 - 5	35	48	35	35
5 - 6	25	57	27	23
6 - 7	24	44	18	23
7 - 8	15	33	24	14
8 - 9	23	29	22	20
9 - 10	15	33	15	7
10 - 11	12	21	16	8
11 - 12	10	22	7	2
12 - 13	10	19	5	4
13 - 14	3	18	6	1
14 - 15	8	12	4	1
15 - 16	1	11	6	
16 - 17	2	6	3	
17 - 18	1	7	4	
18 - 19	3	6	3	
19 - 20	1	4		
20 - 21	1	3	2	
21 - 22	1	3	3	
22 - 23	1	3		
23 - 24		2	1	
24 - 25	1	2	1	
25 - 26		1	1	
26 - 27	2	1	1	
27 - 28		1		
29 - 30	2			
30 - 31		1		
37 - 38			1	
Total	341	519	278	270
Average Energy	6.08	7.85	7.49	4.59

C.M. Spectra of Emitted Particles



Pace style C.M Spectra Table

Plots to show C.M spectra table

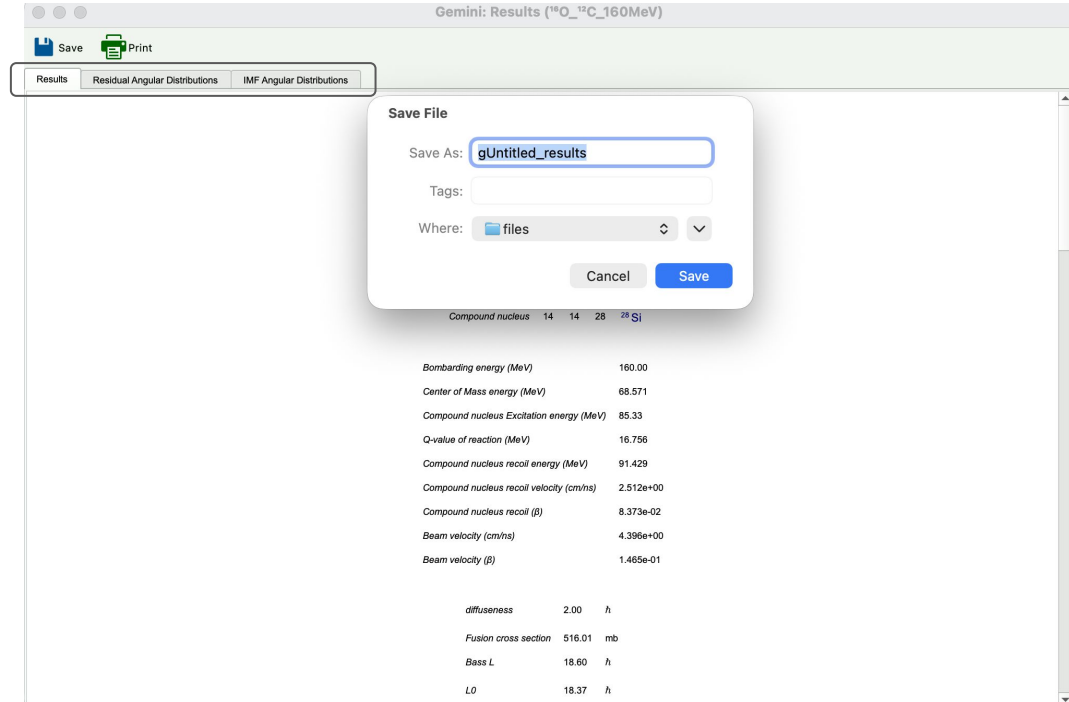
Smaller fixes/Updates

-Tabs for residual angular distributions and IMF angular distributions

-Save button now works as save as first time

-Default save names tweaked

-Scrollbar fixed to always show



Code structure in Gemini v3.3.1

Before

- One serial event loop owned the full cascade
- Global/singleton simulation state was shared by design
- Results were accumulated directly into the UI path

After

- UI gathers run parameters once, then builds event chunks
- Workers run independent CNucleus cascades
- Chunk summaries are merged before the same result views open



Parallelization Applied

Read UI
settings

Build
RunParams

Split into
ChunkJob
ranges

QtConcurrent
worker chunks

Merge
Summary
objects

Code area	Parallel role	Why it matters
gm_mwExecCompound.cpp	CompoundRunParams, CompoundChunkJob, CompoundSummary	Runs compound cascades in independent chunks
gm_mwExecFusion.cpp	FusionRunParams, FusionChunkJob, FusionSummary	Uses the same worker/merge model for fusion
CNucleus + helpers	thread_local products, caches, CEvap, CYrast, RNG	Stops worker threads from sharing mutable decay state
gm_results.cpp	reads GeminiResultRuntimeMs	Shows runtime in the output window, not the main tab

Physics

- No width, barrier, mass, level-density, or evaporation formula was replaced
- Parallel changes move event scheduling and storage ownership, not the decay physics
- Monte Carlo event order is no longer serial, so exact random sequence replay is not expected

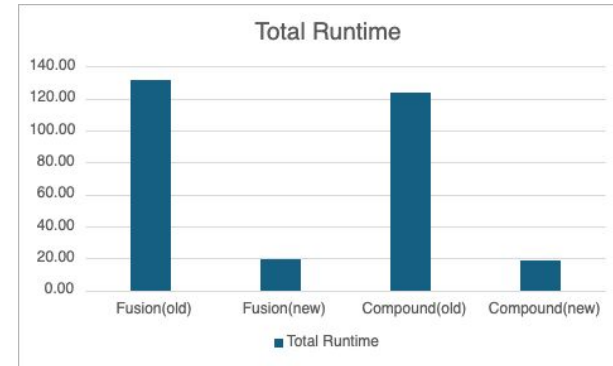
Area	Changed	Preserved
Event execution	Events split across bounded worker chunks	Each event still uses the same CNucleus cascade
Shared state	Mutable state made thread-local per worker	Same per-event calculations and output categories
Compound stability	Evap-channel selection now guards invalid CDF lookup	Valid channel selection follows the existing cumulative-width logic

Speed comes from running independent Monte Carlo events concurrently.

Results

Parallelization produced approximately a 6 $\times$ average speedup in both Fusion and Compound modes.

Mode	projectile			target		compound		Excitation energy @ $Q_{CN}=0$ [MeV]	Events	Previous Runtime	New Runtime	Factor
	A	Z	E [MeV]	A	Z	A	Z					
Fusion	16	8	160	12	6	28	14	68.6	500	7.94	1.2	6.62
	16	8	180	28	14	44	22	114.5	1000	60.43	8.15	7.41
	12	6	120	208	82	220	88	113.5	750	10.33	2.89	3.58
	40	20	200	58	28	98	48	118.4	300	3.00	0.58	5.18
	20	10	150	27	13	47	23	86.2	1500	49.63	6.66	7.45
Compound						198	82	67	300	1.89	0.49	3.86
						40	20	80	500	1.11	0.16	6.95
						60	28	200	750	95.18	12.83	7.42
						120	50	90	1000	11.97	1.73	6.92
						238	92	120	400	13.41	3.42	3.92



Total runtime comparison across all test runs