LHRS Analysis for d_2^n

Radiative Corrections Update: Where Are We?

D. Flay

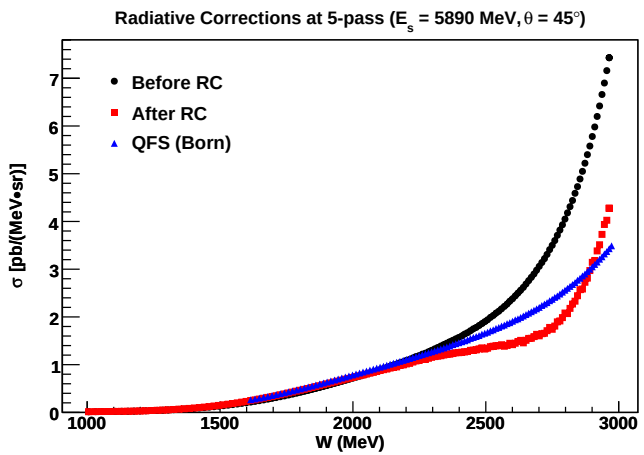
1/19/12

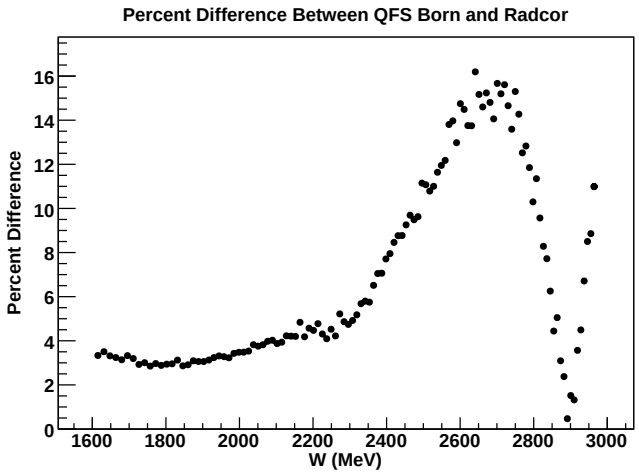
Outline

RADCOR: Fortran Code (1)

Main Problem

- Problem: Unfolded results from RADCOR do not agree with the Born cross section found in QFS at our kinematics (on the order of $\sim 20\%$):

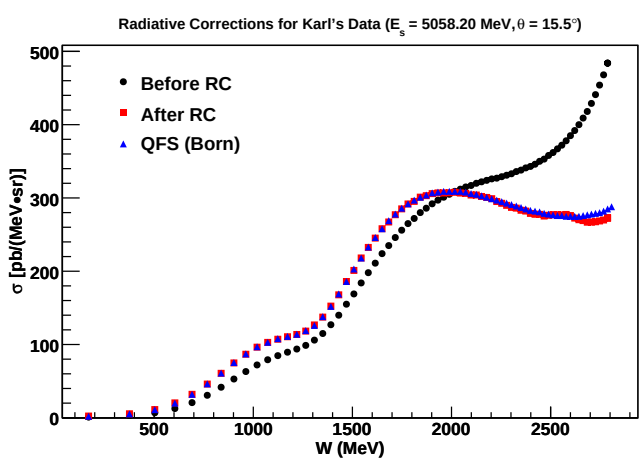


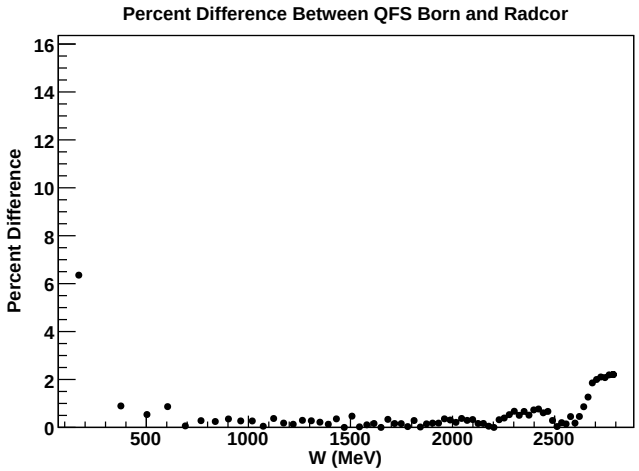


RADCOR: Fortran Code (2)

Main Problem

- But it works for E94-010 kinematics:

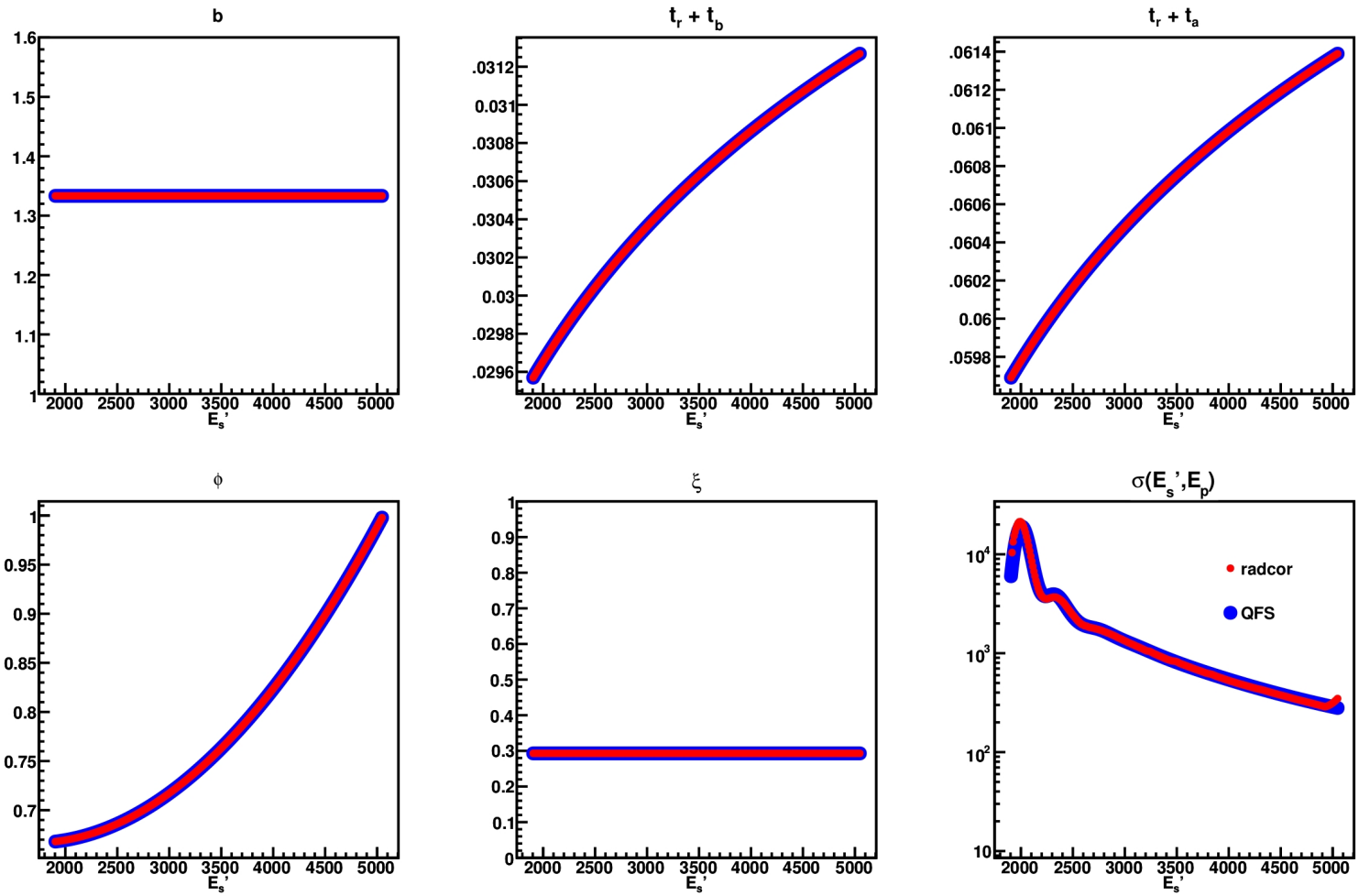




RADCOR: Fortran Code (3)

E94-010: E_s Integrand Decomposition

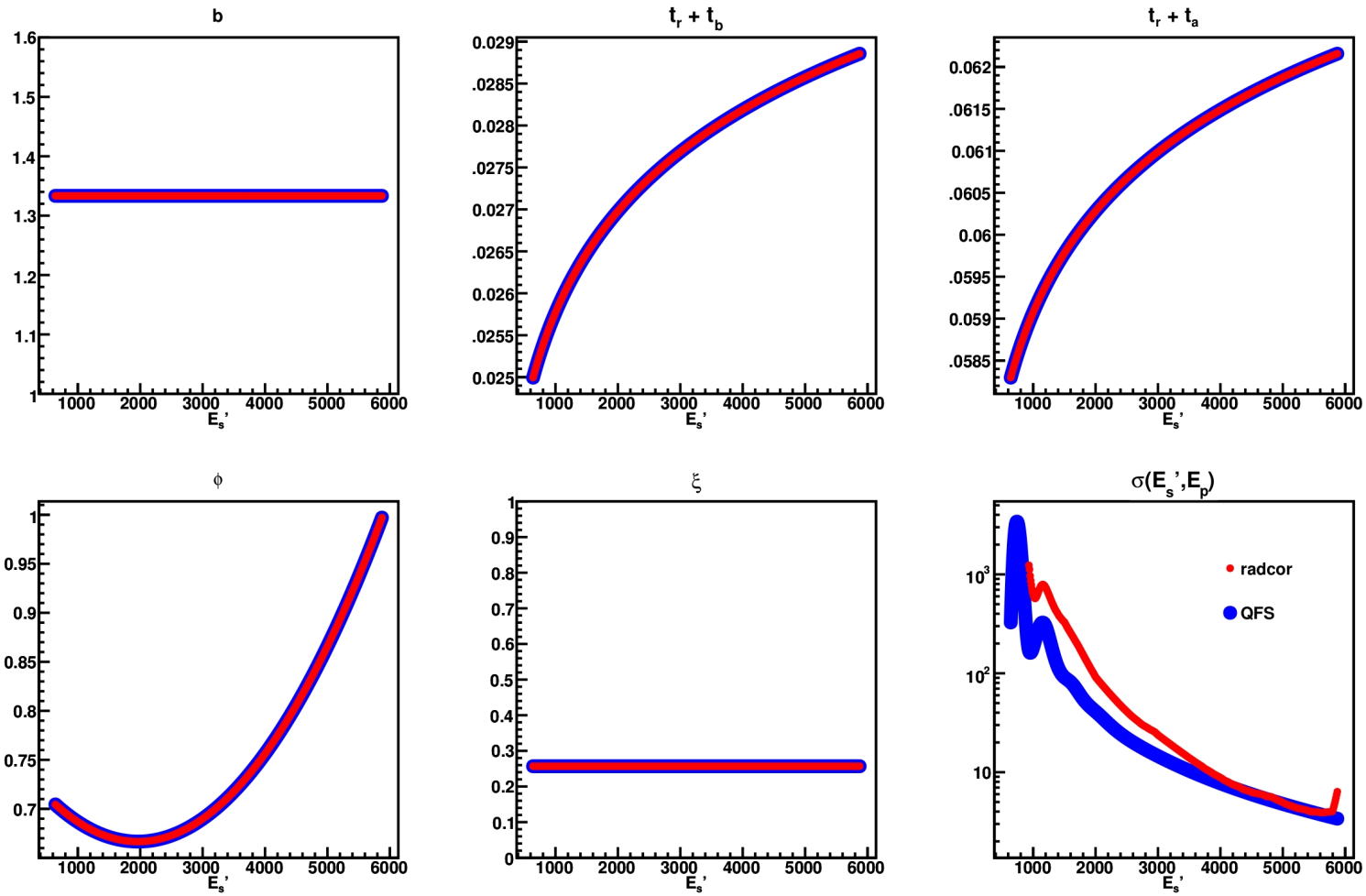
- $E_s = 5058$ MeV, $E_p = 1858$ MeV:



RADCOR: Fortran Code (4)

E06-014: E_s Integrand Decomposition

- $E_s = 5890$ MeV, $E_p = 600$ MeV:



RADCOR: Fortran Code (5)

Discussion

- Differences between the two (input) data sets:
  1. Scattering angle: E94-010 = 15.5°; E06-014 = 45°
  2. QFS free parameters were different for E94-010 and E06-014 (see table)
  3. The QFS model for E06-014 was optimized to fit E94-010, E01-012 and E06-014 data
- Even with optimized parameters for Karl's kinematics, the agreement between QFS Born and the unfolded σ from RADCOR differ by $\sim< 4\%$

Free Parameters for the QFS Model			
Exp	p_F (MeV)	ϵ (MeV)	ϵ_Δ (MeV)
E94-010	220	10	-10
E06-014	130	10	15

C++ Code (1)

Details

- Currently developing C++-based radiative correction code (almost complete)
- Classes:
  1. Spectrum: Holds $\sigma\left(E_p\right)$ for a given E_s and θ. Data members include σ_{Mott}, ν, y and W
  2. RadCor: Calculates all pertinent quantities (integrals, etc.) for a given E_s and E_p
  3. Interpolation: Interpolates cross section data for RadCor
  4. Target: Stores target nucleus info (Z,A), (full) thicknesses (t_b,t_a)
  5. Parameters: Stores miscellaneous parameters from the input file
  6. FileManager: Handles input and output of data
  7. Utilities: Miscellaneous functions that are useful (copying vectors, spectra, etc.)

C++ Code (2)

To Do List

- Full implemenation of code is complete
  - Memory leaks have been completely flushed out (thanks to Valgrind)
  - E_s integrand decomposition shows consistency with both QFS and radcor for the variables b, t_r, t_a, t_b, ϕ and ξ
- Remaining things to do and issues to fix:
  - Check efficiency of code (it takes $\sim$ 30 mins to run one iteration on 5 spectra)
  - Implement a convergence check (was not actually present in RADCOR)
  - Double check interpolation method – cross sections differ from RADCOR
  - Additional interpolation methods
  - Check code against Mo & Tsai, Stein, E94-010 and E01-012 data

Summary

- The problem in RADCOR seems to lie in the cross section interpolation function
- Almost finished development of C++-based code, some things remain to implement

What's Next?

- Radiative Corrections
  - Look to improve the interpolation function in RADCOR
  - Continue development of C++ code