POLYSYS Narrative

Introduction

At its core, the Policy Analysis System (POLYSYS) is structured as a system of interdependent modules simulating (a) county-level crop supply for the continental United States; (b) national crop demands and prices; (c) national livestock supply and demand; and (d) agricultural income. Variables that drive the modules include planted and harvested area, production inputs, yields, exports, costs of production, demand by use, commodity price, government program outlays, and net realized income. Changes in agricultural land use, based on cropland allocation decisions made by individual farmers, are primarily driven by the expected productivity of land, the cost of crop production, the expected economic return on the crop, and market conditions.

POLYSYS compares to other partial equilibrium agricultural policy models reviewed in the following ways:

> FAPRI: This model is also a recursive dynamic model and has deterministic and stochastic versions (NAP, 2011). The spatial resolution is similar as well, covering the continental US, but also accounts for Brazilian agricultural land due to the heavier focus on agricultural trade than POLYSYS currently possesses. This model can link to another model of world commodity markets and Brazil's ethanol market (Pinto & Springer, 2014).
>
> FASOM: The regional and global scales are more course than POLYSYS, although the temporal resolution of daily time steps (Obersteiner, 2012) is more fine. Unlike POLYSYS, this forward-looking model capable of solving all years simultaneously and includes forestland for woody biomass production (NAP, 2011).
>
> REAP: The spatial resolution of this model is similar to POLYSYS, but with all 50 states covered, and likewise uses the USDA Farm Production Regions to define the regional land base (Johansson, et al., 2007).
>
> BEPAM: The temporal scale of this model can be infinitely longer, since this dynamic partial equilibrium model is capable of simulations on a rolling time horizon (Pinto & Springer, 2014).

Primary Routine

Main.for is the main file for POLYSYS and drives all subprocesses, including deterministic and stochastic[1] versions of POLYSYS. An initial call is for the SIM.ins file which is the main driver for all simulations and which holds a series of primarily binary switches related to indices. Index 106 is the primary switch in Main.for to determine which baseline is computed, or subsequently a simulation. When index 106 is determined, a series of other indices (e.g., 41 that calls DataUSDA.txt), follow and control the path the baseline takes through a series of sub-programs or alternatively, the type of simulation. Other important indices called from MAIN.for are: Index 74 that controls a demand or supply simulation, Index 87[2], Index 88 which determines what supply regions are supplied to the model (305 regional LP or 7 regional elasticity), and index 97 that controls the baseline extension[3]. From MAIN.for, and following the call for

---

[1] Stochastic POLYSYS, a 28 year model, has a new spatially-interpolated budgets of traditional crops and energy crops.

[2] When index 106 = 1 or 2, it indicates simulation pathway for baseline runs

[3] Index 97 controls baseline extension: when the number of years in the simulation is over 13 (i.e. when the simulation goes out beyond the USDA baseline), this index is set to 1. Note the USDA baseline is a 10 year baseline, but the 3 additional years are due to lagged or calculated years preceding the baseline.

the sim.ins, a few other program files are called prior to divergent to the different 106 index options: INDX.for that assigns values to an index array, and XTENSION.for[4], as well as OPENFL.for.

Subroutines

Baseline, National

When you are initiating a baseline at the national level (index 106 = 1), the model calls SETUSDA.for as a primary step and computes data from DATAUSDA.txt (index 41 = 0), with data written for verification to ECHOUSDA.txt. At this point the conventional crop information is printed to the Command Prompt terminal. Branching off of SETUSDA.for, SETBIOPROD.for is called to read the USDA baseline data and compute baseline identities with bioproduct costs and technical coefficients read in at this stage.

Livestock module under Index 41 = 0

Following the SETBIOPROD.for step, the livestock module is initiated by calling LIVEBASE.for which computes the baseline for the ERS livestock model and the number of model livestock commodities are calculated[5]. Subsequently, LVREAD.for is called to read the ERS livestock data and writes the baseline matrices to a file after it is finished. SETLVSTK.for is called to read the USDA baseline data and compute the baseline identities for the livestock component. This program file calls the DATALVUSDA.PRN file and livestock information is printed in the Command Prompt terminal at this step. The next two files are also called from LIVEBASE.for and are part of a balancing loop (number of iterations determined by initial stocks and adjustments that need to be made): BASCRTIE.for includes deflated price calculations for feed relevant to the livestock model (e.g., corn, soybean meal) and other livestock calculations (e.g., yearly steer price); BASGCAU.for, which pulls in the feed demand from BASCRTIE.for, is the Grain Consuming Animal Units (GCAU) subroutine that computes the High Protein Consuming Animal Units (HCAU) and the Ruffage Consuming Animal Units (RCAU) programs.

Finishing up Index 41 = 0 with baseline prices and carbon calculations

Once the livestock balancing loop is finished, the model returns to program files called from SETUSDA.for and calls SETDEX.for to compute the baseline prices received and other indices, including the feed price index and livestock production units that were calculated in the livestock subroutine. Subsequently, SETGOVC.for is called to compute the baseline income and expense levels and then SETINC.for computes baseline cash receipts, expenses and government payments if not provided exogenously[6]. Finally, from SETUSDA.for, BASRESID.for opens the cost of residue harvesting data to establish the residue baseline information.

Returning to MAIN.for, and as a result of Index 41 set at 0, SETCARB.for is called to: a) read in base carbon levels by county and by crop (with nasa layering methods), then b) read in base carbon levels nationally weighted (with nasa layering methods), c) read in base carbon levels by county (without nasa layering methods), d) read in base carbon levels nationally weighted (without nasa layering methods), and e) read in carbon accumulation rates by crop.[7] As the final step dictated by Index 41, the ELASTL.for

[4] Defines the extension number that is read in as defined in “simulation number prefix and simulation number suffix (index 39)”. Defines the EXT variable that is used in simulations for linked whatif files and output file naming.
[5] Index 11 is always 1 since ERS model is not maintained and updated external to POLYSYS.
[6] DATAUSDA.xxx file provides this information
[7] This is relevant to carbon accumulation on the land for conventional crops.

file is called to define crop elasticities for testing[8] by reading supply elasticities (1996 supply elasticities) from file ELASTS96.bas and other supply elasticities from ELASTOSP.bas.

Moving onto Index 88 = 0

B2C.for is called from MAIN.for because of the Index 88 defined as 0 in this baseline simulation. At this point the Command Prompt terminal prints "iteration number->1" as this file assigns the first two or more rows (years) of baseline matrices. Since equations require up to 2 lag years (e.g., 2014 and 2015 when you start a simulation in 2016), this step fills in the lag years[9]. Next, WHATIF.for is called so that the user provided information can be supplied to the model. This information calls files housed in the "whatif" folder of the directory in which the executable lies. For the national level baseline simulation, no whatif files are required[10].

Repeated sequence for each year

At this point the Command Prompt terminal prints "year->xxxx" and the model moves into a complex yearly loop as determined by the number of years you have set in the sim.ins file for the simulation. First, SUPPLY2.for[11] is called if you are estimating an extended national baseline over 13[12] years (i.e. beyond the USDA baseline provided exogenously). The Supply Subroutine calculates acreage, yield, production and supply equations for each of the POLYSYS crops. It then defines direct and cross-acreage elasticities for conventional crops, and makes adjustments using coefficients for model crop equations. This subroutine also adjusts baseline acreage so that when it is used in the next year as a lagged value, it will reflect only market changes and not distributed response to a change in diverted acreage. The subroutine finally computes a total of all 8 conventional crop acreages and computes the total unpaid and paid diversions. POLYSYS uses the national supply statements[13] in this baseline simulation to calculate acres, yield, and production and then beginning stocks, total supply identities (defined as the supply minus government stocks that are isolated from the market), and the index of feed prices.

Next, LVSK.for is called to calculate the amount available for domestic consumption from the calculated (livestock subroutine discussed above) production, imports, exports, and removals. Subsequently, LVINIT.for is a "write it" subroutine to set the initial livestock variables to starting values to be used later in LVSIMUL.for. STGAUS is then called to set the endogenous variables to the baseline to start GAUSS.

---

[8] Elasticities for an older version of the livestock module before ERS. We do run through the LVSK.for files. It computes the original livestock information needed for the model but when the ERS version computes livestock information, all of this is overwritten. To be deleted later.

[9] It assigns baseline values to the simulation values for the first two years. Setting those first two years of a simulation equal to the baseline.

[10] For a price simulation, each simulation requires 3 whatif files: whatif.xxxxxx with a matrix housing the offered price by year and by crop, whatifRG.xxxxxx with a matrix of yearly yield improvements, and whatifUS.xxxxxx with a matrix defining the traditional crop yields by year.

[11] Calculates supply for the first iteration of the baseline before going into the LP model. It figures planted and harvested acres from equations. National level corn harvested acres are tied to certain elasticities for predicting corn acreage. At national level it is equal to the baseline for the first 13 years but then for the next n years, this is used.

[12] 3 years of lag as described above.

[13] POLYSYS sets the end of the decision branch on whether to use regional supply or national supply statements to calculate acres, yield, and production.

Balancing loop within the livestock subroutine

First CENTER.for is called from the main index (106=1) and MAIN.for file, and acts as the central simulation program and calls all subprograms of the livestock model (previously computed and discussed above). CENTER.for can utilize CROPTIE.for to do equations that tie the model to the crops if you are utilizing the livestock module and can calculate conventional crop prices. Likewise the following other subprograms can be called by CENTER.for if you are using the livestock module: PRESIM.for, which calculates inventory for livestock and does "pre-simulation" calculations for most of the production equations (i.e. before simulations equations are performed this subprogram sets beginning stocks equal to ending stocks of the year before); LVSIMUL.for to perform simulation iterations for livestock and simultaneously solve for Hogs and Broilers; POSTSIM.for to calculate remaining inventory and prices; GCAU.for to calculate "grain consuming animal units"; and FINISH.for as the final step in this optional livestock subroutine to compile results from the cattle sector for the necessary USDA format.

LVINDEX.for is called at this point to set the index of livestock prices[14] and shares methodology with SETDEX.for. Next, GRAINS.for, the grains subroutine for crop demand equations (uses GAUSS-SIEDEL Routine[15]), is utilized and calculates export demand, feed demand[16], food demand, and industrial demand. Two optional subprograms can be called from GRAINS.for: FLEXGR.for to get the price flexibility schedules for grains and cotton[17] which are based on the ratio of commercial stocks to total use[18]; PRODUCTS.for[19] if using the soybean complex, to integrate agricultural products like soybean oil and meal, as well as sugar into POLYSYS. Returning to MAIN.for, STKCHK.for calculates the stock to use ratio for traditional agricultural uses (food, feed, fiber, fuel, and exports – henceforth FFFF&E). At this point CENTER.for is again called for livestock calculations as well as LVINDEX.for for the index of livestock prices. Subsequently, another loop begins with CENTER.for, then CROPTIE.for if applicable, and so on (see above). This balancing loop lies within a year loop and so occurs yearly and can differ for each iteration as the number of iterations are determined by the initial differences in stocks and prices.

Upon completion of the balancing loop, the model returns to MAIN.for and calls MDLDEX.for to compute the baseline prices received and other indices. MDLGOVC.for tehn computes government payments and REGOUT.for ("regional output") figures the net cash market returns and sums all national numbers to national totals for each category. This is the final step in the national baseline for the initial year; a yearly loop then begins again at SUPPLY2.for at which point the Command Prompt terminal prints "year->xxxx".

---

[14] Even though sheep data are not used to calculate the meat animal index, the weight to construct the livestock price index includes sheep.

[15] This is part of a balancing loop: the weighted average of last iteration and current iteration values of variables are used in the subsequent demand equations; this continues until none of the variable values change appreciably from the previous iteration.

[16] Note: Distillers dry grains from ethanol reduces corn feed demand. One ton of distillers grains substitutes for 35.65 bushels of corn (Delucchi, 1998). Assuming dry grains will sell just a little cheaper than corn for feed, this has the effect of lowering the price of corn, thereby also effecting livestock demand.

[17] This subroutine, if called, will be used each iteration of the grains subroutine: the first iteration baseline values for endogenous demand and price are used (C set to B in STGAUS); predetermined exports are used if specified in the matrix CI4 defined in whatifRG.xxxxxx.

[18] Compute "commercial stock"/ total use ratios and select price flexibility for each feed grain and for wheat.

[19] Values calculated in the IGAUS.for subroutine are brought into the solution algorithm for this calculation.

Baseline, Regional

Following the primary routine and when index 106 is set to 2 (regional baseline), index 41 is determined to be set to 1 ("baseline from baseline RMF") and ELAST.for file is called to define crop elasticities for testing by reading supply elasticities (1996 supply elasticities) from file ELASTS96.bas and other supply elasticities from ELASTOSP.bas.

Just like in the national baseline (index 106=1), when index 106=2 B2C.for is called from MAIN.for. At this point the Command Prompt terminal prints "iteration number->1" as this file assigns the first two or more rows (years) of baseline matrices. Since equations require up to 2 lag years (e.g., 2014 and 2015 when you start a simulation in 2016), this step fills in the lag years. Next, WHATIF.for is called so that the user provided information can be supplied to the model. This information calls files housed in the "whatif" folder of the directory in which the executable lies. For the regional level baseline simulation, no whatif files are required[20].

Index 88 = 1,

RMAIN.for is now called to run a series of subprograms and to drive all internal operations of the supply module. First, READALL.for reads all required data for the baseline and computes all values for the baseline, including the adjustment factors. The only baseline data not computed by READALL.for is the baseline acreage shifts, "BSHIFT(K)", which are computed in INPUT2LP.for. Next READBIO.for is called to read data for biomass crops, and then INFLATE.for figures the inflation rate used in the USDA baseline and this is applied in INFLATEBIO.for. It can also inflate the conventional crops in case the baseline changes (e.g., to FAPRI that has costs in real terms).

Year loop

Next the beginning of a yearly loop (as defined by number of years in simulation set in index 1; up to 28 years) with subroutine RWHATIF.for, but this will be skipped at this stage because it is only used in a simulation. INFLATEBIO.for is called next to inflate biomass harvest costs, production costs, and land rental rates[21]. Inflation indexes used in the USDA baseline to inflate these biomass crop costs[22].

County loop within the year loop

At the beginning of a loop for each of the 3110 counties in the continental United States that POLYSYS handles, BIOHARV.for is called to figure biomass harvest costs and sum the total costs of production. Note that with the 2016 (post-BT16) modification to a 100 year planning horizon, the rotation period is now equal to 100 as well in order to match the planning period in the NPV calculations. COSTCHG.for is called next to figure additional costs from any simulated carbon price, if applicable, and then NPVALUE.for is called to figure the carbon incentive level per acre and determine if either harvest residues or biomass crops take the carbon incentive. If carbon credit is higher than profit associated with the biomass offered price in a supply simulation, then the residues are not harvested. Additional special conditions apply to SRWCs that have an impact on the NPV calculations at this stage: an altered planting schedule is applied to permit even supply across the number of simulated years, based on

---

[20] For a price simulation, each simulation requires 3 whatif files: whatif.xxxxxx with a matrix housing the offered price by year and by crop, whatifRG.xxxxxx with a matrix of yearly yield improvements, and whatifUS.xxxxxx with a matrix defining the traditional crop yields by year.

[21] Land rental rates are increased at 1% per year and are calculated in INFLATEBIO.for but not using the methodology calculated in INFLATE.for

[22] Note: for prices in a simulation, the inflation rates of PPI-CRM[22] were applied in the Billion Ton 2016

rotation of each crop, and any unplanted acres are leased out. Coppice crops (e.g. willows) are on a 4 year incremental planting schedule so that ¼ of the yield of any given acre is carried through to revenue; land rental rates are paid on what is planted and the additional acreage is leased out with all 4 quarters of the land having land rent costs applied by year 4. Non-coppice crops are also on an altered planting schedule on an X year rotation as set in the POPLAR.txt file (e.g., 8 for BT16 analyses) and 1/X of the number of acres in production are then planted in year one with the additional land leased out; by year X, all land is in production and therefore land rent is applicable on all acres.

INPUT2LP.for is the subroutine that sets the maximum and minimum acres before the linear program for each county. It carries out the simulation process, from the calculation of all values for the initial table of the LP, to the computation of all the simulated values for all crops and counties. Next, the LP models are solved in LPSIM.for and the output collected in OUTFRMLP.for[23] to be used at in the subsequent step. Note that DEVYCTY.for can be called at this point to "deviate yields" if the stochastic model is being ran. In addition to capturing the solution values from the LP models, OUTFRMLP.for these values are assigned[24] at this step and baseline shifts are computed to bring back values to the baseline. BIOTRACK.for, a subroutine to track the acres of switchgrass & SRWCs planted but not harvested as well as a balance of any acres in production for these crops from a previous year, is now called and stores[25] the data by POLYSYS region.

An alternative LP model to keep track of other lands besides cropland, cropland in pasture, pastureland, and CRP land[26] can be initiated at this step using PASTLP.for. If utilized, a series of other subprograms are called starting with PASTNPV.for which calculates: the NPV for all energy crops on pasture land, cropland in pasture NPV, and costs and added conversion costs of intensification[27] that are then added to the cost of biomass crops. PASTINLP.for is then called to handle pasture acres in the same way that INPUT2LP.for for energy crops. PASTOUTLP.for subroutine captures the solution values from the LPs and assigns them for pasture acres the same way that OUTFRMP.for assigns solution values for biomass crops.

Expiring CRP acres can be allowed to switch into biomass crops or forestry when index 103 = 1 in the SIM.ins file. If this is turned on, CRPLP.for is called to initiate this alternative LP model and to keep track of other lands besides cropland, cropland in pasture, pastureland, and CRP land as per methodology for PASTLP.for above. Subsequently from CRPLP.for, CRPNPV.for subroutine is called and CRPINLP.for is called and shares methodology with INPUT2LP.for. Next CRPLP.for calls CRPLPSIM.for which contains a linear programming algorithm specifically customized for RASS and Taylor to solve a maximization problem with LT constraints only. The subroutine is also called in INPUT2LP.for. Finally from CRPLP.for, CRPOUTLP.for is called and, like PASTOUTLP.for, shares the same methodology as OUTFRMP.for

[23] This subroutine captures the solution values from LP's and assigns them. It also computes BSHIFTS in baseline to bring values back to baseline
[24] The solution comes out of the LP and the bshifts (adjustment acres) are added back in and then these are applied to the actual planted acres
[25] Keeps track of yields of year planted when multi-year biomass crops are planted
[26] Expiring CRP land can be made available to switch into biomass crops or forestry using index 103=1
[27] Management intensive grazing is controlled by index 114 in the sim.ins file.

Returning to RMAIN.for, HARVEST.for can be called to calculate per unit total costs for each crop and is used for the output file, CTYOUT.txt[28]. As a final step in the loop completed for each county, CARB.for is called to sum the energy and emissions for all acres in each crop and tillage class. This file computes carbon accumulation (soil and afforestation), afforestation and cropland to pasture conversions (added above ground to totals)[29], and costs/benefits to farmers for participation in the program. At this point "BASELINEDONE->x" is printed in the Command Prompt terminal and the model now returns to BIOHARV.for for the next county in this iterative step.

Finishing the year loop

Upon completion of the county loop discussed above, COMPBASE.for is called to compute all regional baseline values, including adjustment factors. COMPIDLE.for can be called from COMPBASE.for to compute totals to estimate adjustment factors after adjusting planted and harvested acres to come to equilibrium with idle land available. At this point the yearly loop starts over again at RWHATIF.for to read and input changes again for the next year.

Final steps in regional baseline and index 87 = 0

Finally, CARB.for is again called from RMAIN.for and the regional baseline for each year is finished off by returning to MAIN.for. GUIWRITE.for is called at this point (index 87=0) and then WOODY.for to consider logging residues, standing wood for harvest, primary mill residues, and urban wood waste. WOODY2.for could be called to handle standing wood data, but is not used in this version of POLYSYS. SUMBIO.for is also called to sum corn grain and woody biomass, including total biomass supply. Next MDLGOVC.for is called to compute government payment and REGOUT.for is called to calculate net cash market returns (total) and sum all national numbers to a national total for each category at the county level.

Simulation

Deterministic or Stochastic

When Index 106 is set to 3 in the sim.ins file, a simulation is started. This must follow computation of the national and regional baselines, although it does not need to immediately follow if a similar scenario preceded the one being ran and baseline files would apply to the simulation in question (e.g., if running base-case at $40 and then at $50). Consistent with the baselines discussed above, MAIN.for begins by opening the Sim.ins file and calls INDX.for and then XTENSION.for and OPENFL.for. Index 99 controls the deterministic (i.e. 1 iteration) or stochastic (100 iterations) versions of the model. A loop is added to return to WHATIF.for as discussed below for stochastic POLYSYS.

Index 41 = 1 (baseline from baseline.rmf)

First, ELAST.for file is called to define crop elasticities for use in determining national variable changes by reading supply elasticities (1996 supply elasticities) from file ELASTS96.bas and other supply elasticities from ELASTOSP.bas. B2C.for is then called B2C.for is called from MAIN.for because of the Index 88 defined as 0 in this baseline simulation. At this point the Command Prompt terminal prints "iteration number->1" as this file assigns the first two or more rows (hyears) of baseline matrices. Since equations require up to 2 lag years (e.g., 2014 and 2015 when you start a simulation in 2016), this step fills in the

---

[28] Since sweet sorghum is an annual crop, the total costs are figured separately for "per dry ton" costs and profit. This is the same for other annual (conventional) crops.

[29] If t=1, cropland to pasture conversion has occurred; if t=2 or 3, afforestation has occurred in this simulation.

lag years[30]. Next, WHATIF.for is called so that the user provided information can be supplied to the model. This information calls files housed in the “whatif” folder of the directory in which the executable lies. Unlike for the baselines discussed above, whatif files should be used to define scenario parameters when appropriate. This subroutine is also the initial step in each stochastic simulation as mentioned above.

Index 74 = 0, demand, and Index 74 = 1, supply simulation

Index 74 in the sim.ins file sets the type of simulation that you want to run: 0 for a demand side simulation, 1 for a supply simulation. You should have associated whatif files when appropriate, but they are not required (e.g., under a AAA000 baseline simulation, no whatifs are needed since it is simulating supply in the absence of a market). For example a price simulation used by DOE required 3 whatif files: whatif.xxxxxx with a matrix housing the offered price by year and by crop, whatifRG.xxxxxx with a matrix of yearly yield improvements and other matrices, and whatifUS.xxxxxx with a matrix defining the traditional crop yields by year.

When Index 74 = 0, BIOPROD1.for is called to see what demand should be, based on price of bioproduct called from whatif.xxxxx as discussed above. This is the only step in a demand simulation that differs from a supply simulation. Subsequently, RMAIN.for is called and the steps below for Index 74 = 1 are followed.

When Index 74 = 1, RMAIN.for is called and sets a series of subroutines to action for the simulation. RWHATIF.for reads and inputs the changes to be simulated first. Next, INFLATEBIO.for inflates biomass harvest costs and production costs, as well as land rental rates.

County loop

Beginning a county loop, BIOHARV.for calculates biomass harvest costs and sums the total costs of production. This subroutine also calculates harvest costs and total costs[31] for switchgrass and other biomass crops. COSTCHG.for also calculates additional costs: from carbon when applicable (i.e. when a carbon credit is applied) and NPVALUE.for is also called to calculate the carbon incentive level per acre when applicable. This subroutine determines if a farmer will either harvest residues at the given price or demand level or if the famer will take the carbon incentive. If the carbon credit (incentive) is higher, then the model forgoes harvest of residues and RESDCNTY.for would then be used, which holds the equations needed to read in and utilize the RUSLE2 data created by David Muth (now of AgSolver, formerly of INL) and bearing his name (i.e. “MUTH2RTBAS.txt”).

INPUT2LP.for carries out the simulation process, from the calculation of all values for the initial tableau of the LP to the computation of all the simulated values for all crops and counties. The LP models are solved in LPSIM.for, which contains the Linear Programming algorithm specifically customized for RASS and Taylor to solve a maximization problem with LT constraints only, and the output collected in OUTFRMLP.for. OUTFRMLP.for also assigns the solution values and computes BSHIFTS in the baseline to bring values back to the baseline. It computes the solution values of the shift in acres from the solution of the LP. These values will always be subtracted to obtain the baseline when running POLYSYS with no

---

[30] It assigns baseline values to the simulation values for the first two years. Setting those first two years of a simulation equal to the baseline.

[31] Note the rotation period used in this subroutine now equals 100 years to match the planning period in the net present value calculations.

changes (i.e. a AAA000 simualtion for a baseline or "business-as-usual"), and also to compute simulation results away from the baseline as is appropriate in a supply or demand simulation. BIOTRACK.for, a subroutine to track the acres of switchgrass (and other energy crops) & SRWCs planted but not harvested as well as a balance of any acres in production for these crops from a previous year, is now called and stores the data by POLYSYS region.

PASTLP.for is an alternative LP model to track other lands besides cropland, cropland in pasture, pastureland, and CRP. This subroutine tracks[32] what land is available for a potential switch into switchgrass, other energy crops, or even forestry. Several additional subroutines are called from PASTLP.for. First, PASTNPV.for is called to calculate NPV for all energy crops on pastureland. Cropland in pasture NPVs are also calculated and this subroutine is responsible for both calculating the general costs but also the costs of conversion for intensification (i.e. MiG) which are added to the costs of the biomass crops. PASTLPSIM.for is called next and contains the same Linear Programming algorithm as LPSIM.for, but customized for pasture acres. Likewise, PASTINLP.for is called next and is modeled after INPUT2LP.for but is modified for pasture acres. Finally, PASTOUTLP.for is called and captures the solution values from LP's and assigns them in the same way discussed above for OUTFRMLP.for but for pasture acres.

Index 103 = 1 for CRP conversion, a final step in the county loop

When expiring CRP land is allowed to convert to energy crops, CRPLP.for is called and contains an alternative LP model to keep track of other lands as described in BIOTRACK.for. This program calls CRPNPV.for to calculate the NPV of land in CRP for the economic decision ofwhat to transition it to when allowed. Note a discount rate of 6.5% is utilized in this program and a planning period of 20 years is applied. Next a subroutine CRPINLP.for is called and follows methodology discussed for INPUT2LP but is modified for pasture acres. Any acres reassigned to CRP in the previous year remain in CRP, and if acres transition to perennial crops they are retained in these crops for the duration of the perennial practice. CRPLPSIM.for is called next and is similar to CRPLPSIM.for, but for CRP acreage. CRPOUTLP.for finishes this step in the county loop and captures the solution values from the LPs and assigns them consistent with PASTOUTLP.for.

Finishing RMAIN.for with Carbon calculations for residues

RESDCNTY.for is called and calls OTHRESDCNTY.for, which computes residue harvesting carbon data when appropriate. CARB.for is then called to sum energy and carbon emissions for all acres in each crop and tillage practice when carbon capture programs are included.

Index 74 = 0, Demand simulation

After going through all subroutines under RMAIN.for, the model returns to MAIN.for and at this point if a demand simulation is used, BIOPROD2.for is called. Subsequently, BIOPROD3 is called.

Index 74 = 1, Supply simulation

After going through all subroutines under RMAIN.for, the model likewise returns to MAIN.for and at this point if a supply simulation is used, WOODY.for is called to consider logging residues, standing wood for harvest, primary mill residues, and urban wood wastes. Next, SUMBIO.for is called to sum all corn grain

[32] A new matrix, PT (J,K,CTY) and PTB (J,K,CTY) is created to keep track of this information. All new switchgrass acres are then transferred to the CY matrix under T=2.T=2 is normally tillage type, but switchgrass tillage is always no-till so it can be utilized for this purpose.

and woody biomass supply, and then LVSK.for, the livestock subroutine, is called to calculate the amount available for domestic consumption from production, imports, exports, and removals. Next, LVINIT.for is called as the “write-it” subroutine to set the initial livestock variables to the starting values. STGAUS.for is called to set the endogenous variables to the baseline and start the GAUSS computations. CENTER.for can then be called and is the central simulation program that calls all subprograms of the livestock model: CROPTIE.for, PRESIM.for, LVSIMUL.for, POSTSIM.for, GCAU.for, and FINISH.for. This is the beginning of a balancing loop when appropriate. The next step in this balancing loop would be LVINDEX.for to set the livestock prices (see SETDEX.for for details) and then GRAINS.for, which calls FLEXGR.for to set the price flexibility schedules for grains and cotton. These price flexibility schedules are based on the ratio of commercial stocks to total use. As part of this balancing loop, this subroutine will be called each iteration of the grains subroutine. STKCHK.for is then called and calls PRODUCTS.for to integrate agricultural prducts like soybean oil and meal into POLYSYS. CENTER.for is then called and finally LVINDEX.for is again called for the index of livestock prices as a final step in the balancing loop.

Index 74 = 0 or 1

MDLDEX.for is called to compute the baseline prices received and other indices, MDLGOVC.for is then called to compute government payments and then REGOUT.for figures net cash market returns and sums all national numbers to a national total for this category. PRINTLOOP.for is called to write regional outputs from the CY matrix and PRINTCRP.for is called to write regional outpurs from the CRP matrix. PRINTRESID.for is called as a final step int eh year loop to print regional outputs from the residue matrix. At this point, the model returns to RMAIN.for to begin the next year in the simulation.

Ending a simulation after all years are completed

After the final iteration (year) in a simulation the following are called: MDLINCC.for, STOCHINI.for, PRINTTXT.for (which calls PAGETXT.for, PAGENO.for, PAGENO2.for), and LUC.for. All of these programs are responsible for the national summary output and are directed by resulttb.fmt for the format of this file (SIMOUT.txt). For stochastic POLYSYS, STOCHINI.for, STATSTOC.for and PRINTSTOC.for are used to run summary statistics on the iterations and to print this information in a human readable format.