10.1071/WF22067
International Journal of Wildland Fire

Supplementary Material

The dynamics of burning activity on degraded peatland in two villages in Central Kalimantan, Indonesia

Samuel Robb[A,*], *Yanetri Asi Nion*[B], *Trisna Anggreini*[B], *Russell Richards*[C,D], *Ammar Abdul Aziz*[E], *Stephen Joseph*[F] and *Paul Dargusch*[A,G]

[A]School of Earth and Environmental Sciences, University of Queensland, St Lucia, Brisbane, Qld 4072, Australia

[B]Faculty of Agriculture, University of Palangka Raya, Palangka Raya, Central Kalimantan, Indonesia

[C]School of Business, University of Queensland, Brisbane, Qld, Australia

[D]Coastal and Marine Research Centre, Griffith University, Gold Coast, Qld, Australia

[E]School of Agriculture and Food Sciences, University of Queensland, Brisbane, Qld, Australia

[F]School of Materials Science and Engineering, University of New South Wales, Sydney, NSW, Australia

[G]Department of Geography, Universitas Negeri Malang, East Java, 65145, Indonesia

*Correspondence to: Email: samuel.robb@uq.net.au

Table S.1: Model Assumptions

Module	Assumption
Population	The population model assumed a simple stock and flow of births, deaths, immigration and emigration. Historical birth and death rates were assumed from Indonesian national statistics (UNPD 2019). Populations were calibrated against survey information (Kemendagri 2015; BRG 2019a). Birth rates continue to converge to two births per woman by 2065. Immigration inflows were assumed as occurring over four years from 1975 to 1979 in Kalampangan.
Labour	The labour module assumed that workers transitioned between livelihoods while seeking to maximise highest long-term (52 week moving average) return to labour. This reflected the attitudes of GMB participants who were receptive to changing livelihoods where higher incomes were available. Livelihoods were assumed to have no barriers to entry where available, however the livelihoods are assumed as being physically and culturally constrained in some scenarios. Physical constraints include limited fish populations based on natural carrying capacity. Cultural constraints include where traditional law prohibits certain practices as with the Dayaknese 'local wisdom' regarding peatland farming, or where only a limited number of alternative livelihoods are available. Labour transitioned between livelihoods by an increment with a maximum threshold per week. Labour increments are assumed in the base case to be no greater than the total labour pool divided by 52, implying

	that the total labour pool may transition between livelihoods over the period of a year. The assumptions of labour increment size and labour allocation response to long-term return to labour averages were not derived from specific sources and were the subject of sensitivity testing.
Fishing	Fisher incomes and trends in fish catch were calibrated against village surveys (TSF 2019), which indicated that fish catch had halved over the prior decade. Actual fish populations were unknown but were selected through calibration against incomes and trends in fish catch, which are variables dependent on fish populations. Fish populations were modelled as a simple stock and flow of births and deaths. Fish populations were constrained by the carrying capacity of the ecosystem. Real fish prices were assumed as being fixed, with local supply having no influence on regional fish prices. There was no affirmation amongst villagers that local fish supply influenced regional prices. Local fishers indicated that fish catch increased threefold during the dry season as low water levels concentrated fish populations (Robb, Nion, Anggreini, Richards, Aziz, et al. 2022). Fish catch per unit of effort therefore included this seasonal fluctuation, with fish yields peaking during the peak of the dry season and reaching a nadir at the peak of the wet season.
Farming	Farming income per hectare of farmland is taken from village level surveys (Kemendagri 2015). Average income per farmer from farming is assumed to be equivalent to the total farmland income divided by the total number of farmers. As farmer density increases, returns to farm labour decline where all else is held constant. Income from farming is assumed to increase at a rate equivalent to Indonesian GDP growth per capita. Farmland demand for every given farmer is initially equivalent to one hectare, this being a median farm size per farmer.

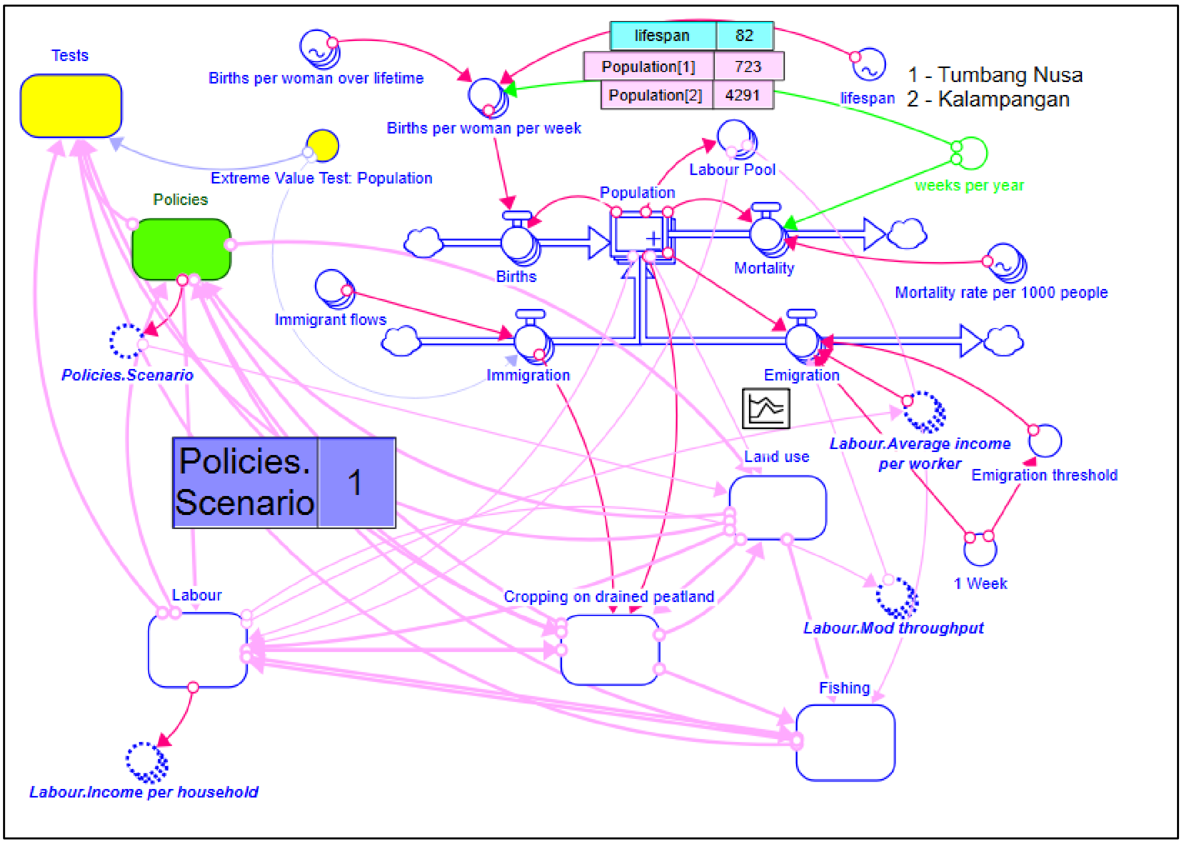
Peatland rate of regeneration	Where peatland is fully rewetted, the rate of regeneration is assumed to be 10% per year, which includes an assumed program of intentional restoration. Where the average peatland water table depth is below zero, the rate of regeneration is assumed to be 1% per year. These rates of regeneration were calibrated against observations from Hościło (2009), where regeneration rates of secondary forest were anticipated to range between 60 years for rarely disturbed degraded peat swamp and several centuries for highly degraded peatland.
Rainfall and climate	Average monthly rainfall data from the model was adjusted for ENSO (El Niño Southern Oscillation) conditions by calibrating against previous models of the relationship between ENSO and rainfall in Palangkaraya adopted from Susilo et al. (2013). Average seasonal rainfall was multiplied by the ENSO index adjusted by a multiplier. As per the averages in Susilo et al. (2013), El Niño decreased rainfall during the dry season, and la Nina conditions increased rainfall during the dry season. Where such conditions prevailed, a multiplier was applied which increased or decreased the influence of the ENSO index. **ENSO Adjusted Weekly Rainfall = Weekly rainfall x (1 – ENSO index × Multiplier** $_{\mathbf{ENSO}}$**)** The multiplier was a constant adjusted as a function of ENSO and seasonal climatic conditions. There were four separate multipliers, El Niño dry season, El Niño wet season, la Nina dry season, la Nina wet season. These multipliers were adjusted so that model rainfall averages for these subcategories reflected averages presented in Susilo et al. (2013) within one standard error.

	The ENSO index itself was taken from historical data (NOAA 2020) in combination with a simple oscillating function with a period of five years and an amplitude of 1.25. $$ENSO(t) = -1.25 \times \sin(\frac{2\pi\,(t + 0.25 \times 26)}{52 \times 5})$$
Fire risk	Wildfire ignition points are assumed to emerge as a result of (i) peatland condition increasing vulnerability to fire and (ii) human population and presence increasing ignition point frequency. A wildfire 'ignition point' is defined as any fire hotspot that occurs outside of a 2 km radius and 48 hour duration of any other prior hotspot. This buffer assumption is adopted from Vetrita and Haryani (2012) to give an estimate of ignition points contained within hotspot datasets. 'Fire risk' is defined as the probability of the occurrence of an ignition point within a defined spatial area and time period (Bachmann and Allgöwer 2001). Peatland condition itself is dependent on land cover, water table depth and seasonal or ENSO climatic conditions. Human causes of hotspots include intentional and unintentional origins. Unintentionally caused fires, such as through haphazard cigarette disposal, are assumed to be proportional to population increases. Intentionally lit fires are assumed as occurring due to (i) controlled fires for agricultural purposes that become uncontrolled, and (ii) uncontrolled fires used for land clearing to prepare land for sale. Intentionally lit ignition points are therefore dependent on labour allocation to the associated livelihoods and land prices. Actual proportional allocations were estimated through calibration against fire hotspot data (fig. 3.4).
Peatland condition	Peatland water table averages were assumed to be a combination of

	drainage through canal building, peat subsidence from organic carbon mineralisation and seasonal rainfall. Physical peat subsidence through initial compaction following drainage and deforestation was not considered. Annual water tables were assumed to rise and fall due to seasonal increases and decreases in average weekly rainfall. Areas that are deforested are assumed to have been drained. Peatland mineralisation was assumed to be a function of peatland water table depth. The relationship between peatland water table depth and greenhouse gas emissions was adopted from Evans et al. (2019), and is defined as the following: $\mathbf{Subs_m = 0.0431\ WTD_m - 1.24}$ (for $WTD_m > 0$) $\mathbf{Subs_m = 0}$ (for $WTD_m \leq 0$) Where 'WTD_m' is the mean water table depth (in centimetres) and '$Subs_m$' is the mean annual subsidence rate (in centimetres). Where WTD_m was less than or equal to zero centimetres, then the rate of subsidence was assumed to be zero.
Greenhouse gas model	Greenhouse gas emissions were measured as a summation of changes to above ground biomass (AGB) carbon stocks and peat soil mineralisation, which included changes in peat soil carbon content. Above ground biomass (AGB) carbon stock estimates were adopted from Hościło (2009), who measured plots of different land cover types within 5 kilometres of the study area. The change in greenhouse gas emissions in a given time period (t-1, t) is calculated in the model as: $\Delta GHG_{t,t-1} = (C_{\mathbf{AGB}}\ (AGB_t - AGB_{t-1}) + C_{\mathbf{PS}}\ (PS_t - PS_{t-1}))\ \delta_{\mathbf{CO_2eq:C}}$ Where $C_{\mathbf{PS}}$ is the carbon content per tonne of peat soil ($MgCO_2e\ Mg^{-1}$);

	C_{AGB} is the carbon content of the above ground biomass (ΔAGB is the change in above ground biomass and ΔPS is the change in peat soil layer due to subsidence . $\delta_{CO_2eq:C}$ is the conversion ratio between organic carbon and carbon dioxide equivalent. Total changes in above ground biomass and peat soil, whether through degradation or accumulation, are calculated as the sum of greenhouse gas flux over time. **Total GHG** = $\sum_1^T C_{\mathbf{AGB}}\,(\mathrm{AGB_t} - \mathrm{AGB_{t-1}})\,\delta_{\mathbf{CO_2eq:C}} + \sum_1^T C_{\mathbf{PS}}(\mathrm{PS_t} - PS_{\mathrm{t-1}})\,\delta_{\mathbf{CO_2eq:C}}$ PS_t =D A_{PS} β Carbon content of dry weight AGB is assumed from Hościło (2009) for woody biomass and Yulianti (2018) for ferns. The carbon content and bulk density of peat soil was assumed as 56% and 0.09 g cm^{-3} respectively as per measurements of peat soil in Central Kalimantan (Hościło 2009).
Biochar yield effect	Biochar was assumed to have a 1.67% increase in saleable harvest yield per tonne of biochar applied per hectare. The aggregate amount of biochar that was assumed to be applied with each planting was 1 Mg ha^{-1}. Biochar was further assumed to be applied in concentrated rates at the planting holes of both crops at an effective rate of 20% weight to weight on pre-rehabilitation cropping and in post-rehabilitation harvestable paludiculture and agroforest products.

Figure S1: SD model - population module



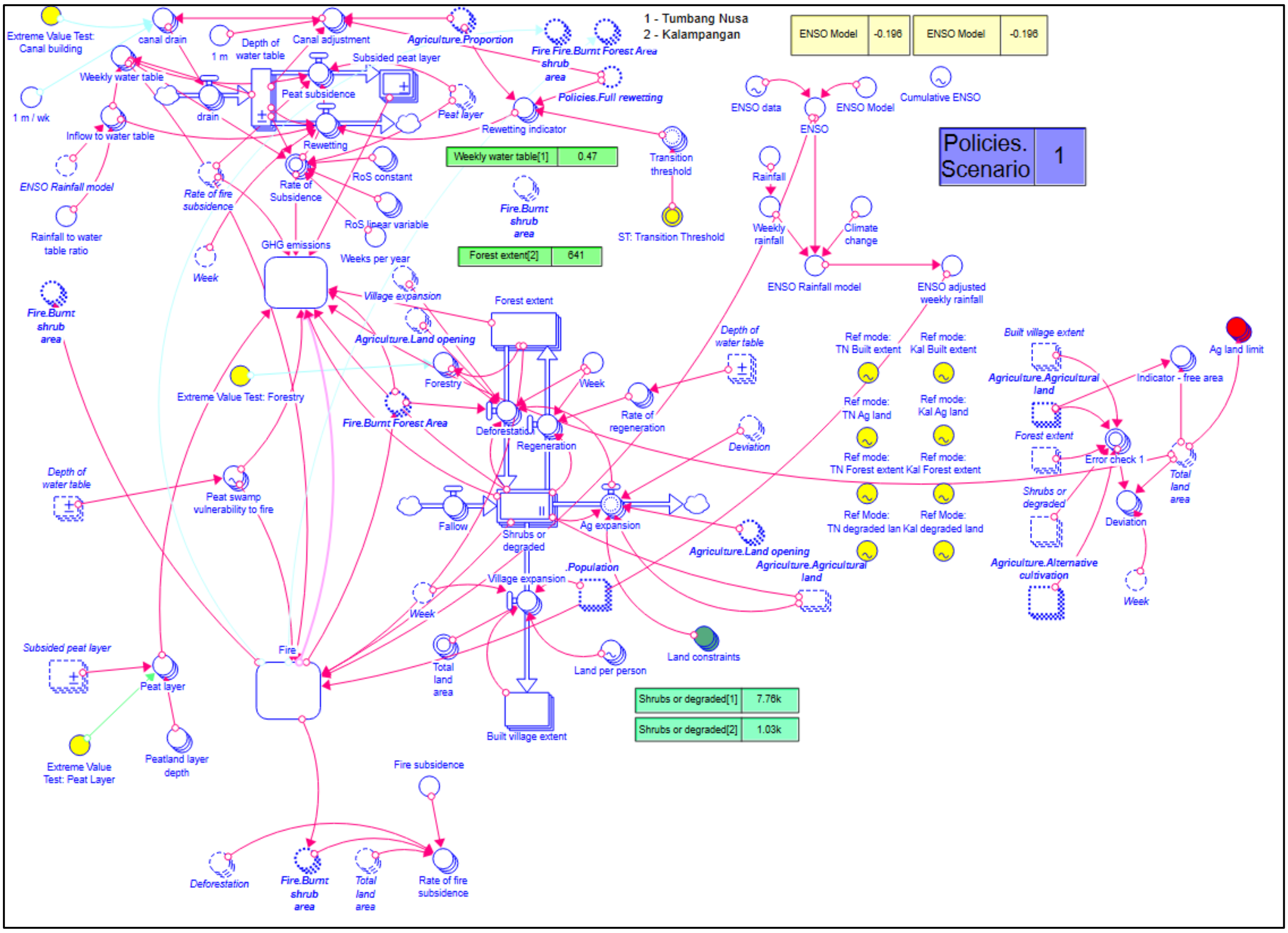


Figure S2: SD model - land use module

Figure S3: SD model - fire module

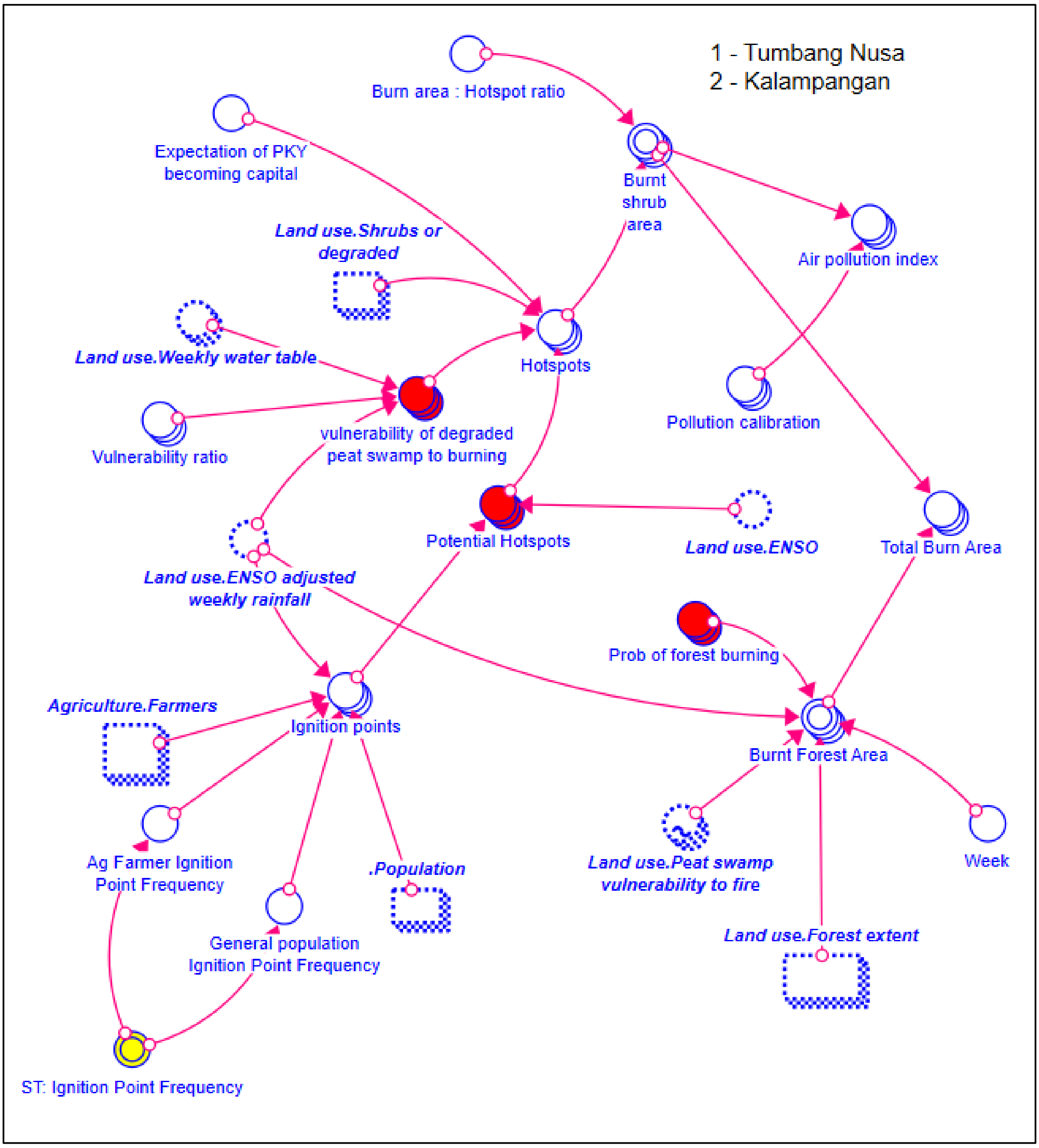


Figure S4: SD model - GHG module

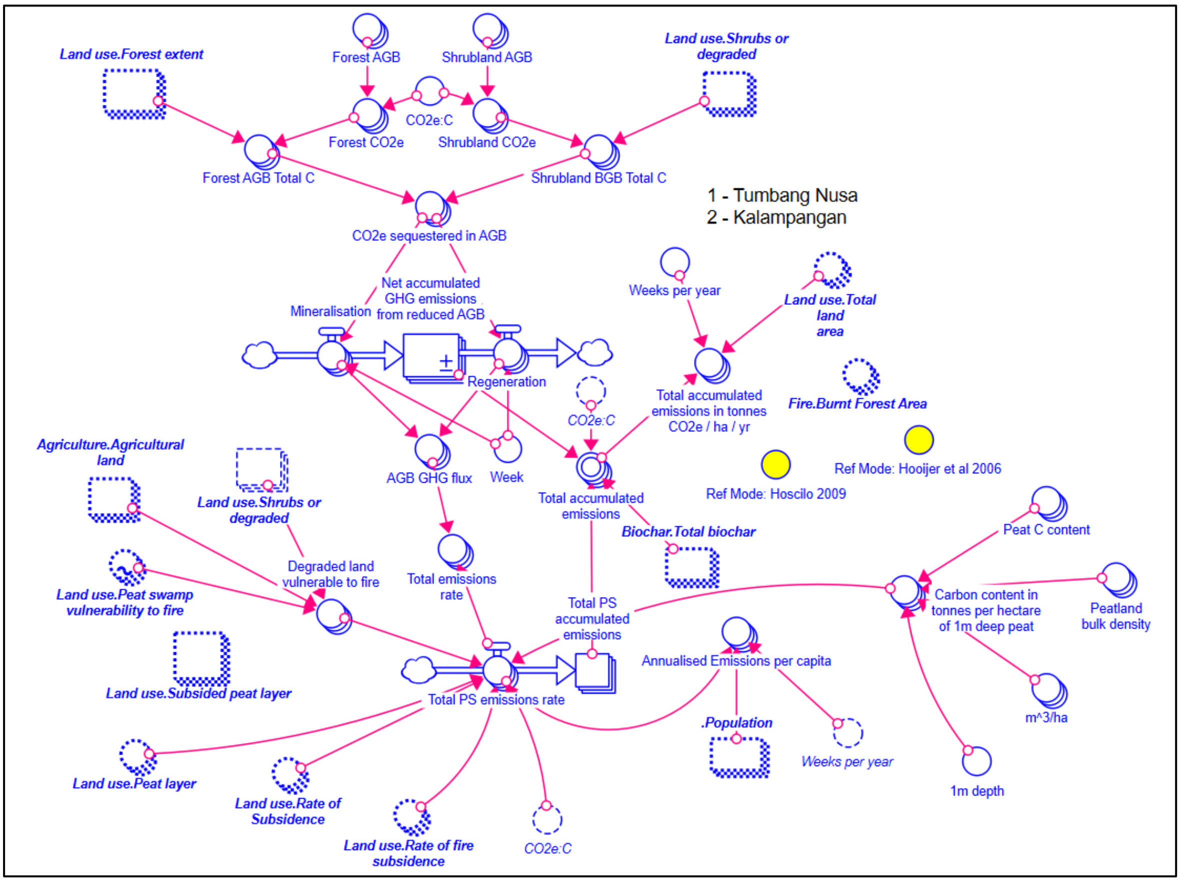
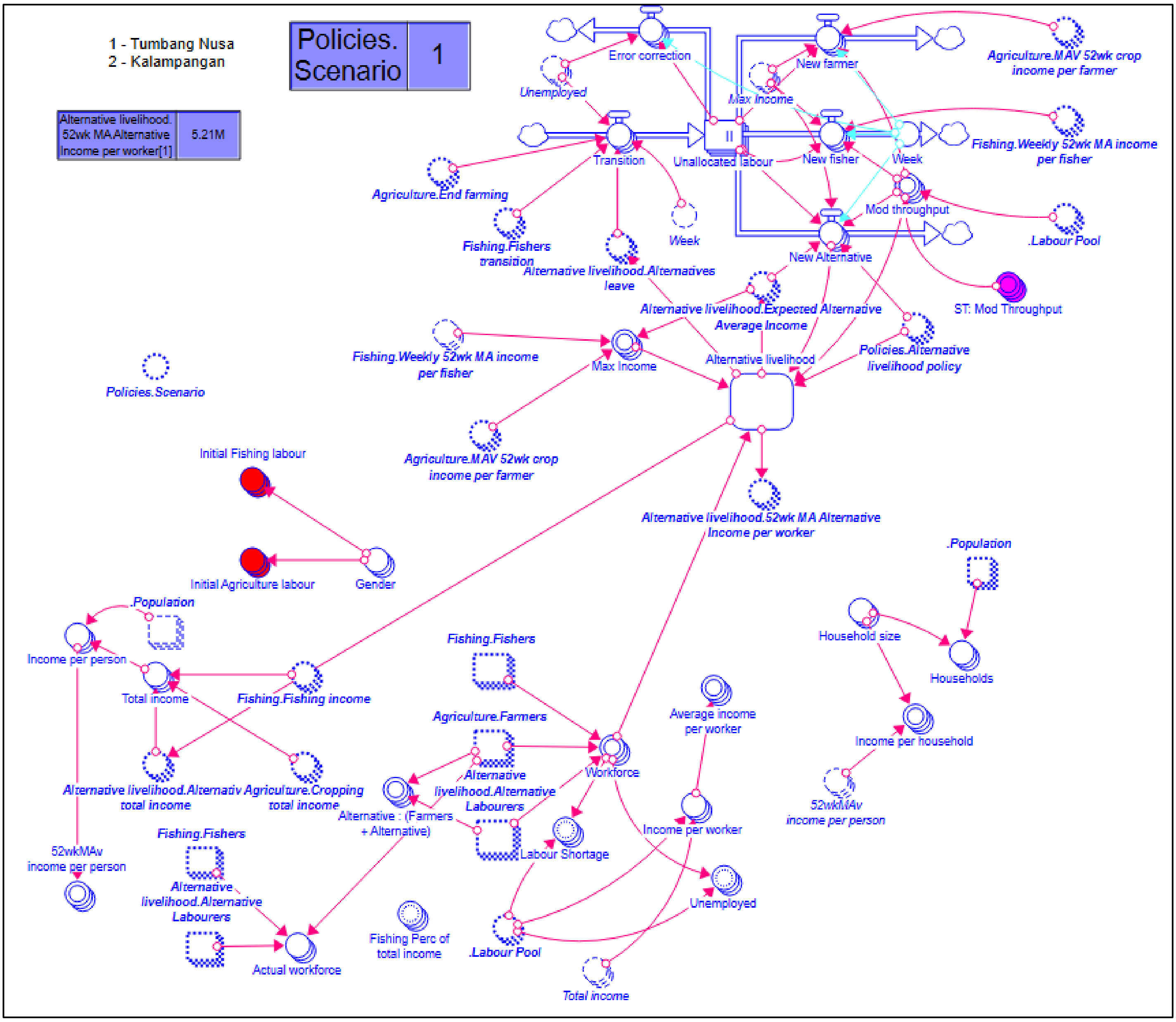


Figure S5: SD model – labour module



1 - Tumbang Nusa
2 - Kalampangan
Policies. Scenario
1
Alternative livelihood. 52wk MA Alternative Income per worker[1]
5.21M
Error correction
Unemployed
New farmer
Agriculture.MAV 52wk crop income per farmer
Max Income
Transition
Unallocated labour
New fisher
Week
Fishing.Weekly 52wk MA income per fisher
Agriculture.End farming
Fishing.Fishers transition
Alternative livelihood.Alternatives leave
Week
Mod throughput
New Alternative
.Labour Pool
ST: Mod Throughput
Alternative livelihood.Expected Alternative Average Income
Policies.Alternative livelihood policy
Fishing.Weekly 52wk MA income per fisher
Max Income
Alternative livelihood
Policies.Scenario
Agriculture.MAV 52wk crop income per farmer
Initial Fishing labour
Alternative livelihood.52wk MA Alternative Income per worker
Initial Agriculture labour
Gender
.Population
.Population
Household size
Households
Income per person
Total income
Fishing.Fishing income
Fishing.Fishers
Agriculture.Farmers
Average income per worker
Income per household
Alternative livelihood.Alternativ total income
Agriculture.Cropping total income
Alternative livelihood.Alternative Labourers
Workforce
52wkMAv income per person
Alternative : (Farmers + Alternative)
Income per worker
52wkMAv income per person
Fishing.Fishers
Labour Shortage
Alternative livelihood.Alternative Labourers
Unemployed
Fishing Perc of total income
Actual workforce
.Labour Pool
Total income

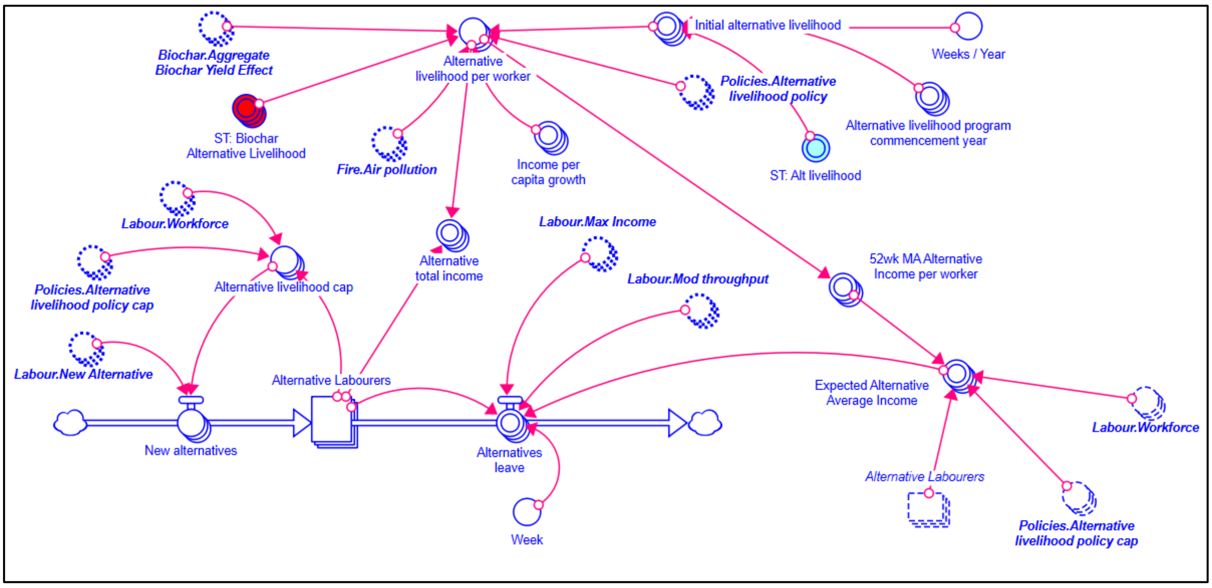
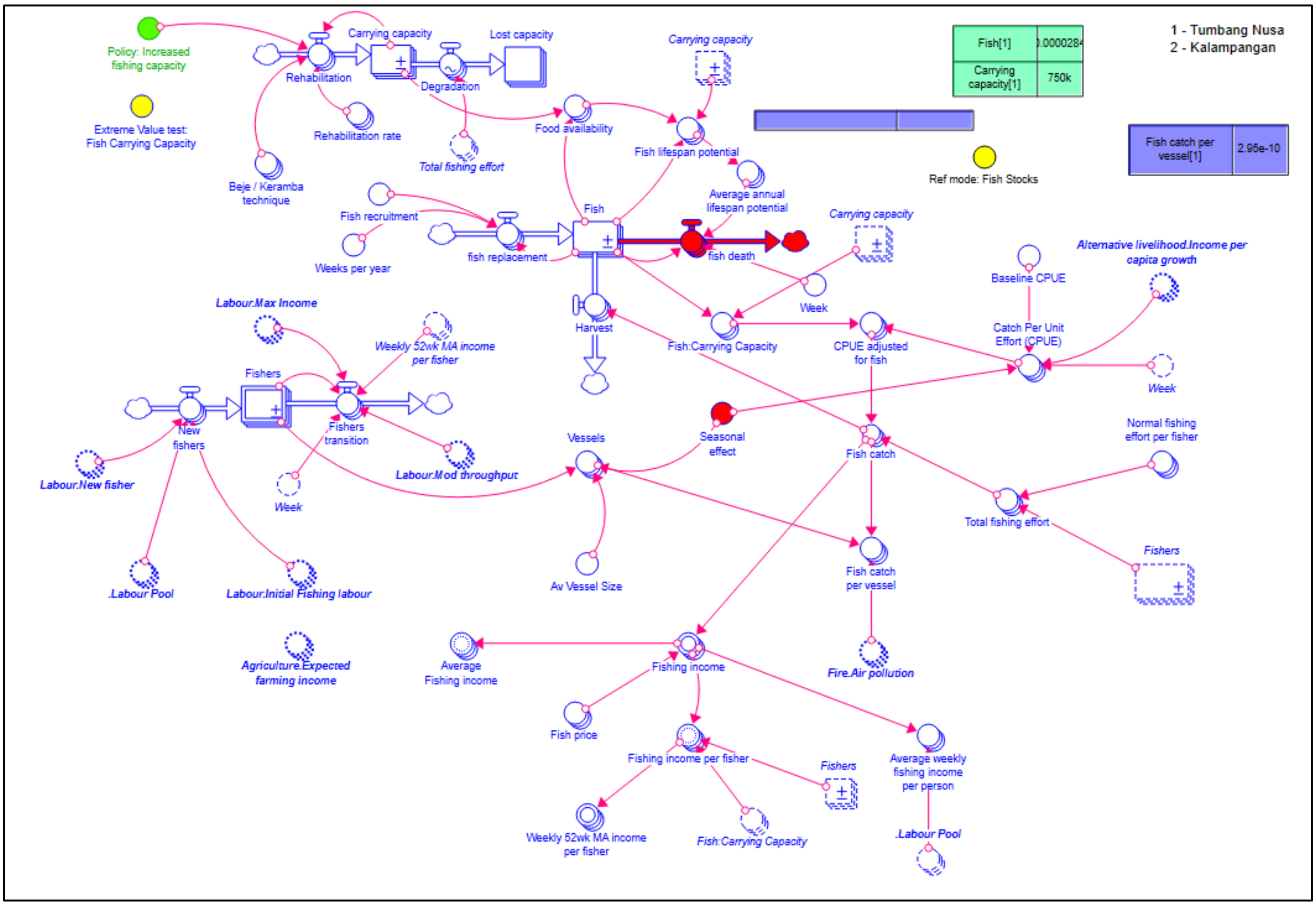


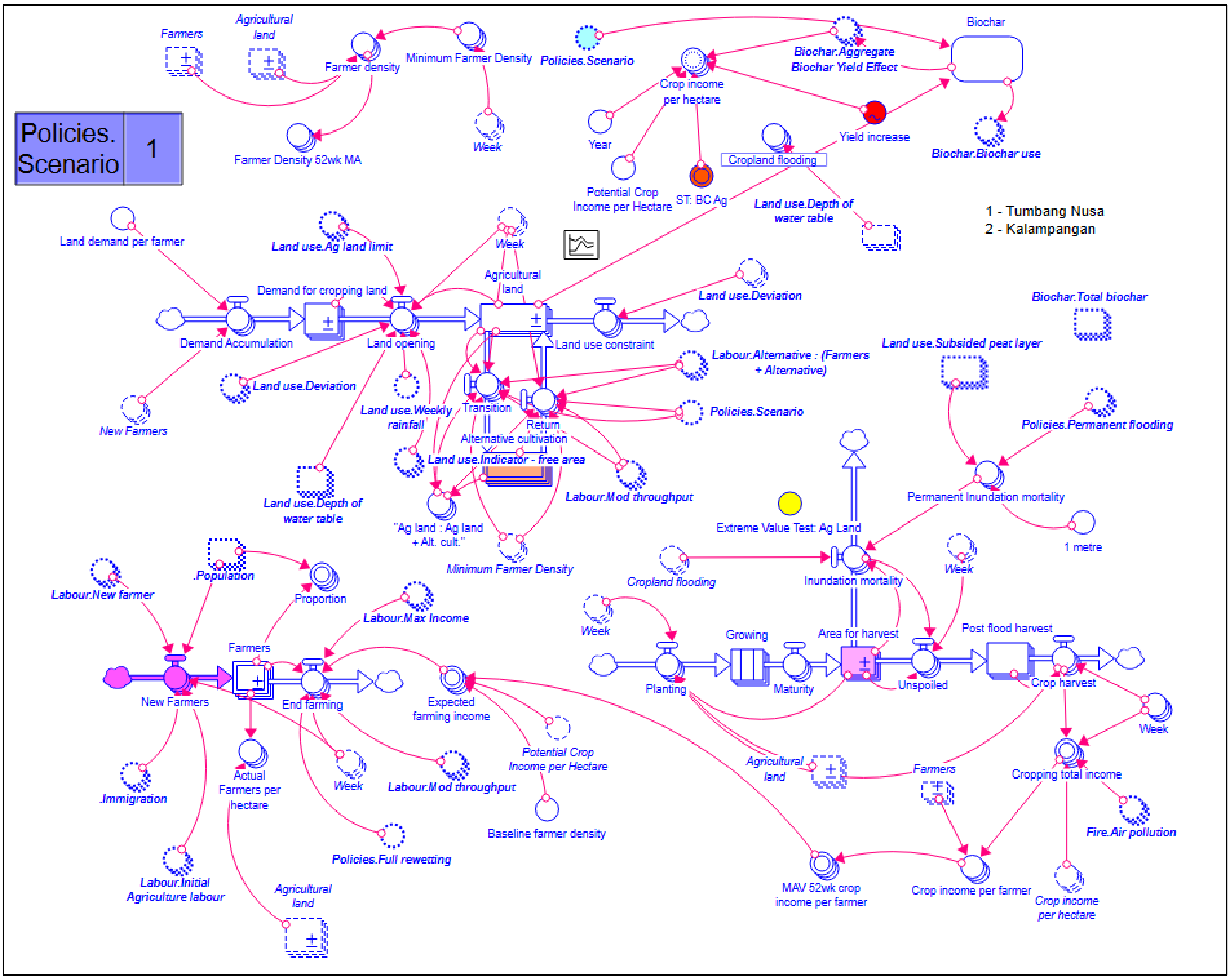
Figure S6: SD model - alternative livelihood module

Figure S7: SD model - fishing module



1 - Tumbang Nusa
2 - Kalampangan
Policy: Increased fishing capacity
Extreme Value test: Fish Carrying Capacity
Carrying capacity
Lost capacity
Rehabilitation
Degradation
Rehabilitation rate
Total fishing effort
Beje / Keramba technique
Food availability
Fish lifespan potential
Average annual lifespan potential
Fish[1]
Carrying capacity[1]
750k
Fish catch per vessel[1]
2.95e-10
Ref mode: Fish Stocks
Fish recruitment
Weeks per year
fish replacement
Fish
fish death
Harvest
Week
Fish:Carrying Capacity
CPUE adjusted for fish
Baseline CPUE
Catch Per Unit Effort (CPUE)
Alternative livelihood.Income per capita growth
Labour.Max Income
Weekly 52wk MA income per fisher
Fishers
New fishers
Fishers transition
Labour.New fisher
Labour.Mod throughput
Vessels
Seasonal effect
Fish catch
Normal fishing effort per fisher
Total fishing effort
.Labour Pool
Labour.Initial Fishing labour
Av Vessel Size
Fish catch per vessel
Agriculture.Expected farming income
Average Fishing income
Fishing income
Fire.Air pollution
Fish price
Fishing income per fisher
Average weekly fishing income per person
Weekly 52wk MA income per fisher

Figure S8: SD model – cropping on drained peatland module



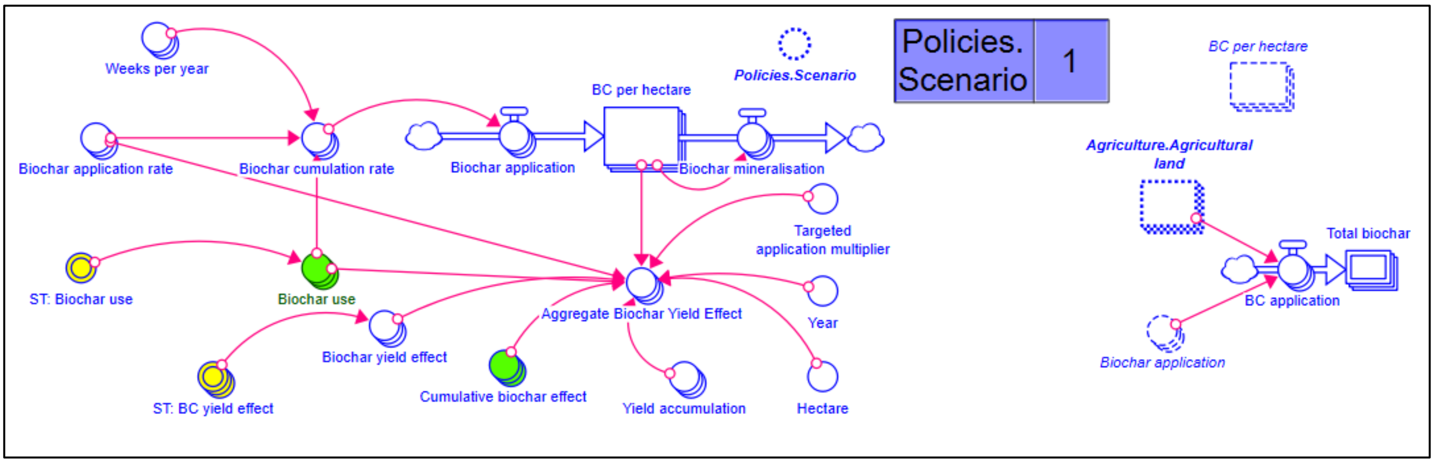


Figure S9: SD model – biochar module

Table S2: Model equations

Top-Level Model:

Population[1](t) = Population[1](t - dt) + (Immigration[1] + Births[1] - Mortality[1] - Emigration[1]) * dt
INIT Population[1] = Extreme_Value_Test:_Population*500
UNITS: People
Population[2](t) = Population[2](t - dt) + (Immigration[2] + Births[2] - Mortality[2] - Emigration[2]) * dt
INIT Population[2] = 0
UNITS: People
4032, Kalampangan 2015
1027, Tumbang Nusa 2018
Births[1] = Births_per_woman_per_week[1]*0.5*Population {UNIFLOW}
UNITS: People/Week
Births[2] = Births_per_woman_per_week[2]*0.5*Population[2] {UNIFLOW}
UNITS: People/Week
Emigration[Villages] = IF(Population <= 0) THEN 0 ELSE IF (Labour.Average_income_per_worker < Emigration_threshold/Week AND Population > 0) THEN Labour.Mod_throughput/Week ELSE 0 {UNIFLOW}
UNITS: People/Week
Immigration[1] = Immigrant_flows[1]*Extreme_Value_Test:_Population/DT {UNIFLOW}
UNITS: People/Week
Immigration[2] = Immigrant_flows[2]*Extreme_Value_Test:_Population/DT {UNIFLOW}
UNITS: People/Week
Mortality[Villages] = IF(Population <= 0) THEN 0 ELSE Population/1000*Mortality_rate_per_1000_people/weeks_per_year {UNIFLOW}
UNITS: People/Week
Births_per_woman_over_lifetime[Villages] = GRAPH(TIME)
UNITS: People/People
Births_per_woman_per_week[Villages] = Births_per_woman_over_lifetime/(lifespan*weeks_per_year)
UNITS: People/People/Week
Emigration_threshold = 200000*(1+.0315/52)^((TIME-50*52)/Week)
UNITS: IDR / Person
Extreme_Value_Test:_Population = 0
UNITS: People / People
Immigrant_flows[1] = 0
UNITS: People/Week
Immigrant_flows[2] = IF(TIME = 15*52) THEN 400 ELSE (IF(TIME = 16*52) THEN 400 ELSE (IF(TIME = 17*52) THEN 400 ELSE (IF(TIME = 18*52) THEN 400 ELSE (IF(TIME = 19*52) THEN 400 ELSE 0))))
UNITS: People/Week
Labour_Pool[Villages] = Population*0.8
UNITS: People
lifespan = GRAPH(TIME)
UNITS: Years
Mortality_rate_per_1000_people[Villages] = GRAPH(TIME)
UNITS: People/People/Year
Week = 1
UNITS: Week
weeks_per_year = 52
UNITS: weeks/year

Agriculture:

Agricultural_land[Villages](t) = Agricultural_land[Villages](t - dt) + (Land_opening[Villages] + Return[Villages] - Land_use_constraint[Villages] - Transition[Villages]) * dt
INIT Agricultural_land[Villages] = 0
UNITS: Hectares
Alternative_cultivation[Villages](t) = Alternative_cultivation[Villages](t - dt) + (Transition[Villages] - Return[Villages]) * dt {NON-NEGATIVE}
INIT Alternative_cultivation[Villages] = 0
UNITS: Hectares
Area_for_harvest[Villages](t) = Area_for_harvest[Villages](t - dt) + (Maturity[Villages] - Inundation_mortality[Villages] - Unspoiled[Villages]) * dt
INIT Area_for_harvest[Villages] = 0
UNITS: Hectares
Demand_for_cropping_land[Villages](t) = Demand_for_cropping_land[Villages](t - dt) + (Demand_Accumulation[Villages] - Land_opening[Villages]) * dt
INIT Demand_for_cropping_land[Villages] = 0
UNITS: Hectares
Farmers[Villages](t) = Farmers[Villages](t - dt) + (New_Farmers[Villages] - End_farming[Villages]) * dt
INIT Farmers[Villages] = 0
UNITS: People
Growing[Villages](t) = Growing[Villages](t - dt) + (Planting[Villages] - Maturity[Villages]) * dt {CONVEYOR}
INIT Growing[Villages] = 0
TRANSIT TIME = 4*3
INFLOW LIMIT = INF
CAPACITY = INF
CONTINUOUS
ACCEPT MULTIPLE BATCHES
UNITS: Hectares
Post_flood_harvest[Villages](t) = Post_flood_harvest[Villages](t - dt) + (Unspoiled[Villages] - Crop_harvest[Villages]) * dt {NON-NEGATIVE}
INIT Post_flood_harvest[Villages] = 0
UNITS: Hectares
Crop_harvest[Villages] = IF(Post_flood_harvest<=0) THEN 0 ELSE Post_flood_harvest/Week {UNIFLOW}
UNITS: Hectares / Week
Demand_Accumulation[Villages] = New_Farmers*Land_demand_per_farmer {UNIFLOW}
UNITS: Hectares/Week
End_farming[Villages] = IF (Farmers <= 0) THEN 0 ELSE IF(Labour.Max_Income > Expected_farming_income) THEN Farmers/Week MOD (Labour.Mod_throughput/Week)*(1+Policies.Full_rewetting) ELSE 0 {UNIFLOW}
UNITS: People/Week
Inundation_mortality[Villages] = IF(Area_for_harvest<=0) THEN 0 ELSE IF(Cropland_flooding=0) THEN Area_for_harvest/DT ELSE IF(Permanent_Inundation_mortality<=0) THEN 0 ELSE Area_for_harvest/DT*Permanent_Inundation_mortality {UNIFLOW}
UNITS: Hectares / Week
Land_opening[Villages] = IF(Land_use.Ag_land_limit<=Agricultural_land) THEN 0 ELSE IF (Land_use.Deviation > 1) THEN 0 ELSE (IF(Agricultural_land >= Land_use.Ag_land_limit) THEN 0 ELSE((IF(Land_use.Depth_of_water_table< 0.4) THEN 0 ELSE (IF(Land_use.Weekly_rainfall <= 0) THEN 0 ELSE Demand_for_cropping_land MOD Land_use.Ag_land_limit)) *Land_use."Indicator_-_free_area"/Week)) {UNIFLOW}
UNITS: Hectares/Week
Land_use_constraint[Villages] = IF (Land_use.Deviation>2) THEN Land_use.Deviation ELSE IF (

Land_use.Ag_land_limit/Week < Agricultural_land/Week) THEN ABS
(Land_use.Ag_land_limit/Week - Agricultural_land/Week) ELSE 0 {UNIFLOW}
UNITS: Hectares/Week
Maturity[Villages] = CONVEYOR OUTFLOW
UNITS: Hectares / Week
New_Farmers[1] = (Labour.New_farmer[1] +IF(TIME=0) THEN
Labour.Initial_Agriculture_labour[1]*.Population[1]/Week ELSE 0) {UNIFLOW}
UNITS: People/Week
New_Farmers[2] = (Labour.New_farmer[2]+.Immigration[2] + IF(TIME=0) THEN
Labour.Initial_Agriculture_labour[2]*.Population[2]/Week ELSE 0) {UNIFLOW}
Planting[Villages] = IF(Area_for_harvest >= Agricultural_land) THEN 0 ELSE
Agricultural_land/Week+Crop_harvest {UNIFLOW}
UNITS: Hectares / Week
Return[Villages] = (IF(Policies.Scenario = 4) THEN 1 ELSE 0)*
(IF(Labour."Alternative_:_(Farmers_+_Alternative)" > "Ag_land_:_Ag_land_+_Alt._cult.") THEN
(Alternative_cultivation/Week MOD Labour.Mod_throughput/Minimum_Farmer_Density/Week)
ELSE 0) {UNIFLOW}
UNITS: Hectares/Week
Transition[Villages] = (IF(Policies.Scenario = 4) THEN 1 ELSE 0)*
(IF(Labour."Alternative_:_(Farmers_+_Alternative)" < "Ag_land_:_Ag_land_+_Alt._cult.") THEN
(Agricultural_land/Week MOD Labour.Mod_throughput/Minimum_Farmer_Density/Week) ELSE 0)
{UNIFLOW}
UNITS: Hectares/Week
Unspoiled[Villages] = Area_for_harvest/Week-Inundation_mortality {UNIFLOW}
UNITS: Hectares / Week
"Ag_land_:_Ag_land_+_Alt._cult."[Villages] = IF(Agricultural_land+Alternative_cultivation <=0)
THEN 0 ELSE Agricultural_land/(Agricultural_land+Alternative_cultivation)
UNITS: Hectare / Hectare
"1_metre" = 1
UNITS: Meter
Actual_Farmers_per_hectare[Villages] = IF(Agricultural_land<=0) THEN 0 ELSE
Farmers/Agricultural_land
UNITS: People/Hectare
Baseline_farmer_density = 1/(0.65)
UNITS: People / Hectare
Crop_income_per_farmer[Villages] = IF(Farmers<=0) THEN 0 ELSE
Cropping_total_income/Farmers
UNITS: IDR / People / Week
Crop_income_per_hectare[Villages] = Potential_Crop_Income_per_Hectare*(1+Yield_increase*1)*
(1+Biochar.Aggregate_Biochar_Yield_Effect*ST:_BC_Ag)*Year
UNITS: IDR/hectare/week
Cropland_flooding[Villages] = IF(Land_use.Depth_of_water_table<0.4) THEN 0 ELSE 1
UNITS: Hectare/Hectare
Cropping_total_income[Villages] = Crop_harvest*Week*Crop_income_per_hectare*
(IF(Fire.Air_pollution_index > 2000) THEN 0.5 ELSE 1)
UNITS: IDR/Week
Expected_farming_income[Villages] = IF(MAV_52wk_crop_income_per_farmer = 0) THEN
Potential_Crop_Income_per_Hectare/Baseline_farmer_density ELSE
MAV_52wk_crop_income_per_farmer
UNITS: IDR / Person / Week
Extreme_Value_Test:_Ag_Land = 1
UNITS: Indicator Variable
Farmer_density[Villages] = IF(Agricultural_land = 0) THEN 0 ELSE   IF(Farmers/Agricultural_land
>= Minimum_Farmer_Density) THEN Farmers/Agricultural_land ELSE Minimum_Farmer_Density
UNITS: People/Hectare

Farmer_Density_52wk_MA[Villages] = MEAN(Farmer_density, DELAY(Farmer_density, 1), DELAY(Farmer_density, 2), DELAY(Farmer_density, 3), DELAY(Farmer_density, 4), DELAY(Farmer_density, 5), DELAY(Farmer_density, 6), DELAY(Farmer_density, 7), DELAY(Farmer_density, 8), DELAY(Farmer_density, 9), DELAY(Farmer_density, 10), DELAY(Farmer_density, 11), DELAY(Farmer_density, 12), DELAY(Farmer_density, 13), DELAY(Farmer_density, 14), DELAY(Farmer_density, 15), DELAY(Farmer_density, 16), DELAY(Farmer_density, 17), DELAY(Farmer_density, 18), DELAY(Farmer_density, 19), DELAY(Farmer_density, 20),   DELAY(Farmer_density, 21), DELAY(Farmer_density, 22), DELAY(Farmer_density, 23), DELAY(Farmer_density, 24), DELAY(Farmer_density, 25), DELAY(Farmer_density, 26), DELAY(Farmer_density, 27), DELAY(Farmer_density, 28), DELAY(Farmer_density, 29), DELAY(Farmer_density, 30), DELAY(Farmer_density, 31), DELAY(Farmer_density, 32), DELAY(Farmer_density, 33), DELAY(Farmer_density, 34), DELAY(Farmer_density, 35), DELAY(Farmer_density, 36), DELAY(Farmer_density, 37), DELAY(Farmer_density, 38), DELAY(Farmer_density, 39), DELAY(Farmer_density, 40), DELAY(Farmer_density, 41), DELAY(Farmer_density, 42), DELAY(Farmer_density, 43), DELAY(Farmer_density, 44), DELAY(Farmer_density, 45), DELAY(Farmer_density, 46), DELAY(Farmer_density, 47), DELAY(Farmer_density, 48), DELAY(Farmer_density, 49), DELAY(Farmer_density, 50),   DELAY(Farmer_density, 51))
UNITS: People/Hectare
Land_demand_per_farmer = 2
UNITS: Hectare / People

MAV_52wk_crop_income_per_farmer[Villages] = MEAN(Crop_income_per_farmer, DELAY(Crop_income_per_farmer, 1), DELAY(Crop_income_per_farmer, 2), DELAY(Crop_income_per_farmer, 3), DELAY(Crop_income_per_farmer, 4), DELAY(Crop_income_per_farmer, 5), DELAY(Crop_income_per_farmer, 6), DELAY(Crop_income_per_farmer, 7), DELAY(Crop_income_per_farmer, 8), DELAY(Crop_income_per_farmer, 9), DELAY(Crop_income_per_farmer, 10), DELAY(Crop_income_per_farmer, 11), DELAY(Crop_income_per_farmer, 12), DELAY(Crop_income_per_farmer, 13), DELAY(Crop_income_per_farmer, 14), DELAY(Crop_income_per_farmer, 15), DELAY(Crop_income_per_farmer, 16), DELAY(Crop_income_per_farmer, 17), DELAY(Crop_income_per_farmer, 18), DELAY(Crop_income_per_farmer, 19), DELAY(Crop_income_per_farmer, 20), DELAY(Crop_income_per_farmer, 21), DELAY(Crop_income_per_farmer, 22), DELAY(Crop_income_per_farmer, 23), DELAY(Crop_income_per_farmer, 24), DELAY(Crop_income_per_farmer, 25), DELAY(Crop_income_per_farmer, 26), DELAY(Crop_income_per_farmer, 27), DELAY(Crop_income_per_farmer, 28), DELAY(Crop_income_per_farmer, 29), DELAY(Crop_income_per_farmer, 30), DELAY(Crop_income_per_farmer, 31), DELAY(Crop_income_per_farmer, 32), DELAY(Crop_income_per_farmer, 33), DELAY(Crop_income_per_farmer, 34), DELAY(Crop_income_per_farmer, 35), DELAY(Crop_income_per_farmer, 36), DELAY(Crop_income_per_farmer, 37), DELAY(Crop_income_per_farmer, 38), DELAY(Crop_income_per_farmer, 39), DELAY(Crop_income_per_farmer, 40), DELAY(Crop_income_per_farmer, 41), DELAY(Crop_income_per_farmer, 42), DELAY(Crop_income_per_farmer, 43), DELAY(Crop_income_per_farmer, 44), DELAY(Crop_income_per_farmer, 45), DELAY(Crop_income_per_farmer, 46), DELAY(Crop_income_per_farmer, 47), DELAY(Crop_income_per_farmer, 48), DELAY(Crop_income_per_farmer, 49), DELAY(Crop_income_per_farmer, 50), DELAY(Crop_income_per_farmer, 51))
UNITS: IDR / Person / Week
Minimum_Farmer_Density[Villages] = 1.0315^(-TIME/52/Week)
UNITS: People / Hectare
Permanent_Inundation_mortality[Villages] = IF(Policies.Permanent_flooding = 0) THEN 0 ELSE (IF(Land_use.Subsided_peat_layer >= 1.5 AND Land_use.Subsided_peat_layer<= 5.5) THEN (Land_use.Subsided_peat_layer-1.5)/4/"1_metre"   ELSE   (IF (Land_use.Subsided_peat_layer < 5.5)

THEN 0 ELSE 1))
UNITS: Metre / Metre
Potential_Crop_Income_per_Hectare = 10^6*1.5
UNITS: IDR/Hectare/Week
Proportion[Villages] = IF (.Population <= 0) THEN 0 ELSE Farmers/.Population
UNITS: People / People
ST:_BC_Ag = 1*1
UNITS: IDR / IDR
Week[Villages] = 1
UNITS: Week
Year = 1
UNITS: Year
Yield_increase = GRAPH(TIME)

UNITS: Tonne/Tonne
Fishing:
Carrying_capacity[1](t) = Carrying_capacity[1](t - dt) + (Rehabilitation[1] - Degradation[1]) * dt
INIT Carrying_capacity[1] = .75*10^6*Extreme_Value_test:_Fish_Carrying_Capacity
UNITS: Fish
Carrying_capacity[2](t) = Carrying_capacity[2](t - dt) + (Rehabilitation[2] - Degradation[2]) * dt
INIT Carrying_capacity[2] = 1*10^4*Extreme_Value_test:_Fish_Carrying_Capacity
UNITS: Fish
Fish[1](t) = Fish[1](t - dt) + (fish_replacement[1] - fish_death[1] - Harvest[1]) * dt
INIT Fish[1] = INIT(Carrying_capacity[1])
UNITS: Fish
Fish[2](t) = Fish[2](t - dt) + (fish_replacement[2] - fish_death[2] - Harvest[2]) * dt
INIT Fish[2] = INIT(Carrying_capacity[2])
UNITS: Fish
Fishers[1](t) = Fishers[1](t - dt) + (New_fishers[1] - Fishers_transition[1]) * dt
INIT Fishers[1] = .Labour_Pool[1]*0.4
UNITS: People
Fishers[2](t) = Fishers[2](t - dt) + (New_fishers[2] - Fishers_transition[2]) * dt
INIT Fishers[2] = 0
UNITS: People
Lost_capacity[Villages](t) = Lost_capacity[Villages](t - dt) + (Degradation[Villages]) * dt {NON-NEGATIVE}
INIT Lost_capacity[Villages] = 0
UNITS: Fish
Degradation[Villages] = GRAPH(Total_fishing_effort)
Points: (0.000, 20.00), (1.000, 0.00) {UNIFLOW, GF EXTRAPOLATED}
UNITS: Fish/Week
fish_death[Villages] = IF(Average_annual_lifespan_potential<=0) THEN Fish/Week ELSE 1/(Average_annual_lifespan_potential)*(Fish) {UNIFLOW}
UNITS: Fish/Week
fish_replacement[Villages] = Fish_recruitment*Fish/Weeks_per_year {UNIFLOW}
UNITS: Fish/Week
Fishers_transition[1] = IF (Fishers <=0) THEN 0 ELSE (IF(Weekly_52wk_MA_income_per_fisher[1]< Labour.Max_Income[1]) THEN Fishers/Week MOD Labour.Mod_throughput[1]/Week ELSE 0) {UNIFLOW}
UNITS: People/Week
Fishers_transition[2] = IF (Fishers <= 0) THEN 0 ELSE (IF(Weekly_52wk_MA_income_per_fisher[2] < Labour.Max_Income[2]) THEN Fishers/Week MOD Labour.Mod_throughput[2]/Week ELSE 0) {UNIFLOW}
UNITS: People/Week
Harvest[Villages] = Fish_catch {UNIFLOW}

UNITS: Fish/Week
New_fishers[Villages] = Labour.New_fisher+ IF(TIME=1) THEN INIT(.Labour_Pool)/DT*Labour.Initial_Fishing_labour ELSE 0 {UNIFLOW}
UNITS: People/Week
Rehabilitation[Villages] = IF(Carrying_capacity < INIT(Carrying_capacity)) THEN (Rehabilitation_rate*Carrying_capacity+"Beje_/_Keramba_technique")/DT ELSE 0 +(IF (TIME > (2020-1965)*52) THEN PULSE(Policy:_Increased_fishing_capacity*0.10*INIT(Carrying_capacity),55*52, 52*5) ELSE 0) {UNIFLOW}
UNITS: Fish/Week
Av_Vessel_Size = 10
UNITS: People/Vessel
Average_annual_lifespan_potential[1] = MEAN(Fish_lifespan_potential[1], DELAY(Fish_lifespan_potential[1],1), DELAY(Fish_lifespan_potential[1],2), DELAY(Fish_lifespan_potential[1],3), DELAY(Fish_lifespan_potential[1],4), DELAY(Fish_lifespan_potential[1],5), DELAY(Fish_lifespan_potential[1],6), DELAY(Fish_lifespan_potential[1],7), DELAY(Fish_lifespan_potential[1],8), DELAY(Fish_lifespan_potential[1],9), DELAY(Fish_lifespan_potential[1],10), DELAY(Fish_lifespan_potential[1],11), DELAY(Fish_lifespan_potential[1],12), DELAY(Fish_lifespan_potential[1],13), DELAY(Fish_lifespan_potential[1],14), DELAY(Fish_lifespan_potential[1],15), DELAY(Fish_lifespan_potential[1],16), DELAY(Fish_lifespan_potential[1],17), DELAY(Fish_lifespan_potential[1],18), DELAY(Fish_lifespan_potential[1],19), DELAY(Fish_lifespan_potential[1],20), DELAY(Fish_lifespan_potential[1],21), DELAY(Fish_lifespan_potential[1],22), DELAY(Fish_lifespan_potential[1],23), DELAY(Fish_lifespan_potential[1],24), DELAY(Fish_lifespan_potential[1],25), DELAY(Fish_lifespan_potential[1],26), DELAY(Fish_lifespan_potential[1],27), DELAY(Fish_lifespan_potential[1],28), DELAY(Fish_lifespan_potential[1],29), DELAY(Fish_lifespan_potential[1],30), DELAY(Fish_lifespan_potential[1],31), DELAY(Fish_lifespan_potential[1],32), DELAY(Fish_lifespan_potential[1],33), DELAY(Fish_lifespan_potential[1],34), DELAY(Fish_lifespan_potential[1],35), DELAY(Fish_lifespan_potential[1],36), DELAY(Fish_lifespan_potential[1],37), DELAY(Fish_lifespan_potential[1],38), DELAY(Fish_lifespan_potential[1],39), DELAY(Fish_lifespan_potential[1],40), DELAY(Fish_lifespan_potential[1],41), DELAY(Fish_lifespan_potential[1],42), DELAY(Fish_lifespan_potential[1],43), DELAY(Fish_lifespan_potential[1],44), DELAY(Fish_lifespan_potential[1],45), DELAY(Fish_lifespan_potential[1],46), DELAY(Fish_lifespan_potential[1],47), DELAY(Fish_lifespan_potential[1],48), DELAY(Fish_lifespan_potential[1],49), DELAY(Fish_lifespan_potential[1],50), DELAY(Fish_lifespan_potential[1],51), DELAY(Fish_lifespan_potential[1],52))
UNITS: Weeks
Average_annual_lifespan_potential[2] = MEAN(Fish_lifespan_potential[2], DELAY(Fish_lifespan_potential[2],1), DELAY(Fish_lifespan_potential[2],2), DELAY(Fish_lifespan_potential[2],3), DELAY(Fish_lifespan_potential[2],4), DELAY(Fish_lifespan_potential[2],5), DELAY(Fish_lifespan_potential[2],6), DELAY(Fish_lifespan_potential[2],7), DELAY(Fish_lifespan_potential[2],8), DELAY(Fish_lifespan_potential[2],9), DELAY(Fish_lifespan_potential[2],10), DELAY(Fish_lifespan_potential[2],11), DELAY(Fish_lifespan_potential[2],12), DELAY(Fish_lifespan_potential[2],13), DELAY(Fish_lifespan_potential[2],14), DELAY(Fish_lifespan_potential[2],15), DELAY(Fish_lifespan_potential[2],16), DELAY(Fish_lifespan_potential[2],17), DELAY(Fish_lifespan_potential[2],18), DELAY(Fish_lifespan_potential[2],19), DELAY(Fish_lifespan_potential[2],20), DELAY(Fish_lifespan_potential[2],21), DELAY(Fish_lifespan_potential[2],22), DELAY(Fish_lifespan_potential[2],23), DELAY(Fish_lifespan_potential[2],24), DELAY(Fish_lifespan_potential[2],25), DELAY(Fish_lifespan_potential[2],26), DELAY(Fish_lifespan_potential[2],27), DELAY(Fish_lifespan_potential[2],28),

DELAY(Fish_lifespan_potential[2],29), DELAY(Fish_lifespan_potential[2],30),
DELAY(Fish_lifespan_potential[2],31), DELAY(Fish_lifespan_potential[2],32),
DELAY(Fish_lifespan_potential[2],33), DELAY(Fish_lifespan_potential[2],34),
DELAY(Fish_lifespan_potential[2],35), DELAY(Fish_lifespan_potential[2],36),
DELAY(Fish_lifespan_potential[2],37), DELAY(Fish_lifespan_potential[2],38),
DELAY(Fish_lifespan_potential[2],39), DELAY(Fish_lifespan_potential[2],40),
DELAY(Fish_lifespan_potential[2],41), DELAY(Fish_lifespan_potential[2],42),
DELAY(Fish_lifespan_potential[2],43), DELAY(Fish_lifespan_potential[2],44),
DELAY(Fish_lifespan_potential[2],45), DELAY(Fish_lifespan_potential[2],46),
DELAY(Fish_lifespan_potential[2],47), DELAY(Fish_lifespan_potential[2],48),
DELAY(Fish_lifespan_potential[2],49), DELAY(Fish_lifespan_potential[2],50),
DELAY(Fish_lifespan_potential[2],51), DELAY(Fish_lifespan_potential[2],52))
UNITS: Weeks
Average_Fishing_income[Villages] = MEAN(Fishing_income, DELAY(Fishing_income,1),
DELAY(Fishing_income,2), DELAY(Fishing_income,3), DELAY(Fishing_income,4),
DELAY(Fishing_income,5), DELAY(Fishing_income,6), DELAY(Fishing_income,7),
DELAY(Fishing_income,8), DELAY(Fishing_income,9), DELAY(Fishing_income,10),
DELAY(Fishing_income,11), DELAY(Fishing_income,12), DELAY(Fishing_income,13),
DELAY(Fishing_income,14), DELAY(Fishing_income,15), DELAY(Fishing_income,16),
DELAY(Fishing_income,17), DELAY(Fishing_income,18), DELAY(Fishing_income,19),
DELAY(Fishing_income,20), DELAY(Fishing_income,21), DELAY(Fishing_income,22),
DELAY(Fishing_income,23), DELAY(Fishing_income,24), DELAY(Fishing_income,25),
DELAY(Fishing_income,26), DELAY(Fishing_income,27), DELAY(Fishing_income,28),
DELAY(Fishing_income,29), DELAY(Fishing_income,30), DELAY(Fishing_income,31),
DELAY(Fishing_income,32), DELAY(Fishing_income,33), DELAY(Fishing_income,34),
DELAY(Fishing_income,35), DELAY(Fishing_income,36), DELAY(Fishing_income,37),
DELAY(Fishing_income,38), DELAY(Fishing_income,39), DELAY(Fishing_income,40),
DELAY(Fishing_income,41), DELAY(Fishing_income,42), DELAY(Fishing_income,43),
DELAY(Fishing_income,44), DELAY(Fishing_income,45), DELAY(Fishing_income,46),
DELAY(Fishing_income,47), DELAY(Fishing_income,48), DELAY(Fishing_income,49),
DELAY(Fishing_income,50), DELAY(Fishing_income,51), DELAY(Fishing_income,52))
UNITS: IDR/Week
Average_weekly_fishing_income_per_person[Villages] = IF(.Labour_Pool <=0) THEN 0 ELSE
Fishing_income/.Labour_Pool
UNITS: IDR/Person/Week
Baseline_CPUE = 7
UNITS: Fish/Effort
"Beje_/_Keramba_technique"[Villages] = 0
UNITS: Fish
"Catch_Per_Unit_Effort_(CPUE)"[Villages] = Baseline_CPUE*
(1+Alternative_livelihood.Income_per_capita_growth/52/2)^(IF(TIME>52*30) THEN (TIME-
52*30)/Week ELSE 0) *Seasonal_effect
UNITS: Fish/Effort
CPUE_adjusted_for_fish[Villages] = "Catch_Per_Unit_Effort_(CPUE)"*Fish:Carrying_Capacity
UNITS: Fish/Effort
Extreme_Value_test:_Fish_Carrying_Capacity = 1
UNITS: Fish
Fish_catch[Villages] = Total_fishing_effort*CPUE_adjusted_for_fish
UNITS: Fish/Week
Fish_catch_per_vessel[Villages] = (IF(Fire.Air_pollution_index > 2000) THEN 1-0.25 ELSE
IF(Fire.Air_pollution_index > 1000) THEN 1-0.1 ELSE 1)* IF(Vessels <= 0.1) THEN 0 ELSE (
Fish_catch/Vessels /52)
UNITS: Fish/Vessel/Week
Fish_lifespan_potential[1] = IF (Fish>Carrying_capacity[1]) THEN (3*52
{weeks}*(Food_availability[1])) ELSE 3*52{weeks}

UNITS: Weeks
Fish_lifespan_potential[2] = IF (Fish>Carrying_capacity[2]) THEN (3*52*(Food_availability[2])) ELSE 3*52
UNITS: Weeks
Fish_price[Villages] = 100000
UNITS: IDR/Fish
Fish_recruitment = 1.75/2
UNITS: Fish / Fish / Year
Fish:Carrying_Capacity[Villages] = (Fish)/(IF(Carrying_capacity = 0) THEN 0.000000001 ELSE Carrying_capacity)
UNITS: Fish / Fish
Fishing_income[Villages] = Fish_price*Fish_catch
UNITS: IDR/Weeks
Fishing_income_per_fisher[Villages] = IF(Fishers>0) THEN Fishing_income/(Fishers) ELSE IF (Fish:Carrying_Capacity >1.5) THEN 1.5*10^6 ELSE 0
UNITS: IDR/People/Week
Food_availability[1] = IF(Fish[1]>Carrying_capacity[1]) THEN Carrying_capacity[1]/(Fish[1]) ELSE 1
UNITS: Fish / Fish
Food_availability[2] = IF(Fish[2]>Carrying_capacity[2]) THEN Carrying_capacity[2]/Fish[2] ELSE 1
UNITS: Fish / Fish
Normal_fishing_effort_per_fisher[Villages] = 1
UNITS: Effort / Person / Week
Policy:_Increased_fishing_capacity = 0
UNITS: Fish / Fish
Ref_mode:_Fish_Stocks = 0
UNITS: Fish
Rehabilitation_rate[Villages] = 1/52
UNITS: Fish / Fish
Seasonal_effect = SIN((TIME+0.25*26)*PI/26)+1
UNITS: Fish / Fish
Total_fishing_effort[Villages] = Normal_fishing_effort_per_fisher*Fishers
UNITS: Effort/Week
Vessels[Villages] = Fishers/Av_Vessel_Size*(Seasonal_effect/2*0.75+0.25)
UNITS: Vessel
Week = 1
UNITS: Week
Weekly_52wk_MA_income_per_fisher[Villages] = MEAN(Fishing_income_per_fisher, DELAY(Fishing_income_per_fisher,1), DELAY(Fishing_income_per_fisher,2), DELAY(Fishing_income_per_fisher,3), DELAY(Fishing_income_per_fisher,4), DELAY(Fishing_income_per_fisher,5), DELAY(Fishing_income_per_fisher,6), DELAY(Fishing_income_per_fisher,7), DELAY(Fishing_income_per_fisher,8), DELAY(Fishing_income_per_fisher,9), DELAY(Fishing_income_per_fisher,10), DELAY(Fishing_income_per_fisher,11), DELAY(Fishing_income_per_fisher,12), DELAY(Fishing_income_per_fisher,13), DELAY(Fishing_income_per_fisher,14), DELAY(Fishing_income_per_fisher,15), DELAY(Fishing_income_per_fisher,16), DELAY(Fishing_income_per_fisher,17), DELAY(Fishing_income_per_fisher,18), DELAY(Fishing_income_per_fisher,19), DELAY(Fishing_income_per_fisher,20), DELAY(Fishing_income_per_fisher,21), DELAY(Fishing_income_per_fisher,22), DELAY(Fishing_income_per_fisher,23), DELAY(Fishing_income_per_fisher,24), DELAY(Fishing_income_per_fisher,25), DELAY(Fishing_income_per_fisher,26), DELAY(Fishing_income_per_fisher,27), DELAY(Fishing_income_per_fisher,28), DELAY(Fishing_income_per_fisher,29), DELAY(Fishing_income_per_fisher,30), DELAY(Fishing_income_per_fisher,31), DELAY(Fishing_income_per_fisher,32),

DELAY(Fishing_income_per_fisher,33), DELAY(Fishing_income_per_fisher,34),
DELAY(Fishing_income_per_fisher,35), DELAY(Fishing_income_per_fisher,36),
DELAY(Fishing_income_per_fisher,37), DELAY(Fishing_income_per_fisher,38),
DELAY(Fishing_income_per_fisher,39), DELAY(Fishing_income_per_fisher,40),
DELAY(Fishing_income_per_fisher,41), DELAY(Fishing_income_per_fisher,42),
DELAY(Fishing_income_per_fisher,43), DELAY(Fishing_income_per_fisher,44),
DELAY(Fishing_income_per_fisher,45), DELAY(Fishing_income_per_fisher,46),
DELAY(Fishing_income_per_fisher,47), DELAY(Fishing_income_per_fisher,48),
DELAY(Fishing_income_per_fisher,49), DELAY(Fishing_income_per_fisher,50),
DELAY(Fishing_income_per_fisher,51), DELAY(Fishing_income_per_fisher,52))
UNITS: IDR/People/Week
Weeks_per_year = 52
UNITS: Weeks / Year

Labour:
Unallocated_labour[Villages](t) = Unallocated_labour[Villages](t - dt) + (Transition[Villages] - New_farmer[Villages] - New_fisher[Villages] - New_Alternative[Villages] - Error_correction[Villages]) * dt {NON-NEGATIVE}
INIT Unallocated_labour[Villages] = 0
UNITS: People
Error_correction[Villages] = IF(Unallocated_labour>Unemployed) THEN (Unallocated_labour-Unemployed)/Week ELSE 0 {UNIFLOW}
OUTFLOW PRIORITY: 4
UNITS: People/Week
New_Alternative[Villages] = Policies.Alternative_livelihood_policy* IF(Alternative_livelihood.Expected_Alternative_Average_Income>= Max_Income) THEN (Unallocated_labour MOD Mod_throughput*12)/Week ELSE 0 {UNIFLOW}
OUTFLOW PRIORITY: 3
UNITS: People/Week
New_farmer[Villages] = IF(Max_Income<= Agriculture.MAV_52wk_crop_income_per_farmer) THEN (Unallocated_labour MOD Mod_throughput)/Week ELSE 0 {UNIFLOW}
OUTFLOW PRIORITY: 1
UNITS: People/Week
New_fisher[Villages] = IF(Max_Income <= Fishing.Weekly_52wk_MA_income_per_fisher) THEN Unallocated_labour/Week MOD Mod_throughput/Week ELSE 0 {UNIFLOW}
OUTFLOW PRIORITY: 2
UNITS: People/Week
Transition[Villages] = IF(TIME > 1) THEN (Fishing.Fishers_transition+Alternative_livelihood.Alternatives_leave+Agriculture.End_farming) + Unemployed/Week ELSE 0 {UNIFLOW}
UNITS: People/Week
"52wkMAv_income_per_person"[Villages] = MEAN(Income_per_person , DELAY(Income_per_person ,1), DELAY(Income_per_person ,2), DELAY(Income_per_person ,3), DELAY(Income_per_person ,4), DELAY(Income_per_person ,5), DELAY(Income_per_person ,6), DELAY(Income_per_person ,7), DELAY(Income_per_person ,8), DELAY(Income_per_person ,9), DELAY(Income_per_person ,10), DELAY(Income_per_person ,11), DELAY(Income_per_person ,12), DELAY(Income_per_person ,13), DELAY(Income_per_person ,14), DELAY(Income_per_person ,15), DELAY(Income_per_person ,16), DELAY(Income_per_person ,17), DELAY(Income_per_person ,18), DELAY(Income_per_person ,19), DELAY(Income_per_person ,20), DELAY(Income_per_person ,21), DELAY(Income_per_person ,22), DELAY(Income_per_person ,23), DELAY(Income_per_person ,24), DELAY(Income_per_person ,25), DELAY(Income_per_person ,26), DELAY(Income_per_person ,27), DELAY(Income_per_person ,28), DELAY(Income_per_person ,29), DELAY(Income_per_person ,30), DELAY(Income_per_person ,31), DELAY(Income_per_person ,32), DELAY(Income_per_person

,33), DELAY(Income_per_person ,34), DELAY(Income_per_person ,35), DELAY(Income_per_person ,36), DELAY(Income_per_person ,37), DELAY(Income_per_person ,38), DELAY(Income_per_person ,39), DELAY(Income_per_person ,40), DELAY(Income_per_person ,41), DELAY(Income_per_person ,42), DELAY(Income_per_person ,43), DELAY(Income_per_person ,44), DELAY(Income_per_person ,45), DELAY(Income_per_person ,46), DELAY(Income_per_person ,47), DELAY(Income_per_person ,48), DELAY(Income_per_person ,49), DELAY(Income_per_person ,50), DELAY(Income_per_person ,51))
UNITS: IDR / Person / Week
Actual_workforce[Villages] = Agriculture.Farmers+Alternative_livelihood.Alternative_Labourers+Fishing.Fishers
UNITS: People
"Alternative_:_(Farmers_+_Alternative)"[Villages] = IF(Agriculture.Farmers+Alternative_livelihood.Alternative_Labourers<=0) THEN 0 ELSE (Agriculture.Farmers/(Agriculture.Farmers+Alternative_livelihood.Alternative_Labourers))
UNITS: People / People
Average_income_per_worker[Villages] = MEAN(Income_per_worker , DELAY(Income_per_worker ,1), DELAY(Income_per_worker ,2), DELAY(Income_per_worker ,3), DELAY(Income_per_worker ,4), DELAY(Income_per_worker ,5), DELAY(Income_per_worker ,6), DELAY(Income_per_worker ,7), DELAY(Income_per_worker ,8), DELAY(Income_per_worker ,9), DELAY(Income_per_worker ,10), DELAY(Income_per_worker ,11), DELAY(Income_per_worker ,12), DELAY(Income_per_worker ,13), DELAY(Income_per_worker ,14), DELAY(Income_per_worker ,15), DELAY(Income_per_worker ,16), DELAY(Income_per_worker ,17), DELAY(Income_per_worker ,18), DELAY(Income_per_worker ,19), DELAY(Income_per_worker ,20), DELAY(Income_per_worker ,21), DELAY(Income_per_worker ,22), DELAY(Income_per_worker ,23), DELAY(Income_per_worker ,24), DELAY(Income_per_worker ,25), DELAY(Income_per_worker ,26), DELAY(Income_per_worker ,27), DELAY(Income_per_worker ,28), DELAY(Income_per_worker ,29), DELAY(Income_per_worker ,30), DELAY(Income_per_worker ,31), DELAY(Income_per_worker ,32), DELAY(Income_per_worker ,33), DELAY(Income_per_worker ,34), DELAY(Income_per_worker ,35), DELAY(Income_per_worker ,36), DELAY(Income_per_worker ,37), DELAY(Income_per_worker ,38), DELAY(Income_per_worker ,39), DELAY(Income_per_worker ,40), DELAY(Income_per_worker ,41), DELAY(Income_per_worker ,42), DELAY(Income_per_worker ,43), DELAY(Income_per_worker ,44), DELAY(Income_per_worker ,45), DELAY(Income_per_worker ,46), DELAY(Income_per_worker ,47), DELAY(Income_per_worker ,48), DELAY(Income_per_worker ,49), DELAY(Income_per_worker ,50), DELAY(Income_per_worker ,51))
UNITS: IDR / Person / Week
Fishing_Perc_of_total_income[1] = 0.9
UNITS: IDR / IDR
Fishing_Perc_of_total_income[2] = 0.1
UNITS: IDR / IDR
Gender[Villages] = 1
UNITS: People/People
Household_size[Villages] = 4
UNITS: People/Household
Households[Villages] = .Population/Household_size
UNITS: Household
Income_per_household[Villages] = "52wkMAv_income_per_person"*Household_size
UNITS: IDR / Household / Week
Income_per_person[Villages] = IF(.Population<=0) THEN 0 ELSE Total_income/.Population
UNITS: IDR / Person / Week

Income_per_worker[Villages] = IF (.Labour_Pool <= 0) THEN 0 ELSE Total_income/.Labour_Pool
UNITS: IDR / Person / Week
Initial_Agriculture_labour[1] = 0.1*Gender
UNITS: People/People
Initial_Agriculture_labour[2] = 0.9*Gender
UNITS: People/People
Initial_Fishing_labour[1] = 0.8*Gender
UNITS: People / People
Initial_Fishing_labour[2] = 0.1*Gender
UNITS: People / People
Labour_Shortage[Villages] = IF(Workforce>=.Labour_Pool) THEN Workforce-.Labour_Pool ELSE 0
UNITS: People
Max_Income[Villages] = MAX(MAX(Agriculture.MAV_52wk_crop_income_per_farmer, Fishing.Weekly_52wk_MA_income_per_fisher), MAX(Alternative_livelihood.Expected_Alternative_Average_Income, Agriculture.MAV_52wk_crop_income_per_farmer))
UNITS: IDR / People / Week
Mod_throughput[1] = ST:_Mod_Throughput* IF(.Labour_Pool[1]<100) THEN 10 ELSE .Labour_Pool[1]/520
UNITS: People
Mod_throughput[2] = ST:_Mod_Throughput *IF(.Labour_Pool[2]<100) THEN 30 ELSE .Labour_Pool[2]/520
UNITS: People
ST:_Mod_Throughput[Villages] = 1*1
UNITS: People / People
Total_income[Villages] = Agriculture.Cropping_total_income+ Fishing.Fishing_income+ Alternative_livelihood.Alternative_total_income
UNITS: IDR/Week
Unemployed[Villages] = IF(.Labour_Pool>Workforce) THEN ABS(.Labour_Pool-Workforce) ELSE 0
UNITS: People
Week = 1
UNITS: Week
Workforce[1] = Fishing.Fishers[1]+Agriculture.Farmers[1]+Alternative_livelihood.Alternative_Labourers[1]
UNITS: People
Workforce[2] = Fishing.Fishers[2]+Agriculture.Farmers[2]+Alternative_livelihood.Alternative_Labourers[2]
UNITS: People

Land_use:
Built_village_extent[1](t) = Built_village_extent[1](t - dt) + (Village_expansion[1]) * dt {NON-NEGATIVE}
INIT Built_village_extent[1] = INIT(.Population)*Land_per_person
UNITS: Hectares
Built_village_extent[2](t) = Built_village_extent[2](t - dt) + (Village_expansion[2]) * dt {NON-NEGATIVE}
INIT Built_village_extent[2] = 0
UNITS: Hectares
Depth_of_water_table[Villages](t) = Depth_of_water_table[Villages](t - dt) + (drain[Villages] - Peat_subsidence[Villages] - Rewetting[Villages]) * dt
INIT Depth_of_water_table[Villages] = 0
UNITS: Meter
Forest_extent[Villages](t) = Forest_extent[Villages](t - dt) + (Regeneration[Villages] -

Deforestation[Villages]) * dt {NON-NEGATIVE}
INIT Forest_extent[Villages] = Total_land_area-INIT(Built_village_extent)
UNITS: Hectares
Shrubs_or_degraded[Villages](t) = Shrubs_or_degraded[Villages](t - dt) + (Deforestation[Villages] + Fallow[Villages] - Regeneration[Villages] - Ag_expansion[Villages] - Village_expansion[Villages]) * dt {NON-NEGATIVE}
INIT Shrubs_or_degraded[Villages] = 0
UNITS: Hectares
Subsided_peat_layer[Villages](t) = Subsided_peat_layer[Villages](t - dt) + (Peat_subsidence[Villages]) * dt
INIT Subsided_peat_layer[Villages] = 0
UNITS: Meter
Ag_expansion[Villages] = IF (Deviation > 0 OR Agriculture.Agricultural_land < Land_constraints AND Shrubs_or_degraded > 0)   THEN Agriculture.Land_opening+Deviation ELSE 0 {UNIFLOW}
OUTFLOW PRIORITY: 2
UNITS: Hectares/Week
Deforestation[Villages] = IF(Shrubs_or_degraded<=0 OR Ag_expansion>0) THEN Ag_expansion +Village_expansion+Fire.Burnt_Forest_Area+ (Forestry)/Week ELSE Fire.Burnt_Forest_Area+(Forestry)/Week +Agriculture.Land_opening {UNIFLOW}
UNITS: Hectares/Week
drain[Villages] = (canal_drain-DELAY(canal_drain, DT)) {UNIFLOW}
UNITS: Meters/Week
Fallow[Villages] = 0 {UNIFLOW}
UNITS: Hectares/Week
Peat_subsidence[1] = IF (Peat_layer[1] <=0) THEN 0 ELSE   (IF(Weekly_water_table[1]>0) THEN (Rate_of_Subsidence[1]+Rate_of_fire_subsidence[1]) ELSE 0) {UNIFLOW}
UNITS: Meters/Week
Peat_subsidence[2] = IF (Peat_layer[2] <= 0) THEN 0 ELSE   (IF(Weekly_water_table[2]>0) THEN (Rate_of_Subsidence[2]+Rate_of_fire_subsidence[2]) ELSE 0) {UNIFLOW}
UNITS: Meters/Week
Regeneration[Villages] = IF (Shrubs_or_degraded > 0) THEN Shrubs_or_degraded*Rate_of_regeneration/Week ELSE 0 {UNIFLOW}
OUTFLOW PRIORITY: 1
UNITS: Hectares/Week
Rewetting[Villages] = IF (Rewetting_indicator = 1 AND Depth_of_water_table > 0) THEN Depth_of_water_table/Week MOD Inflow_to_water_table/Week ELSE 0 {UNIFLOW}
UNITS: Meters/Week
Village_expansion[Villages] = IF(Built_village_extent>Total_land_area) THEN 0 ELSE (.Population*Land_per_person-DELAY(.Population*Land_per_person, DT))/Week {UNIFLOW}
OUTFLOW PRIORITY: 3
UNITS: Hectares/Week
"1_m" = 1
UNITS: Meter
"1_m_/_wk" = 1
UNITS: Meter/Week
Ag_land_limit[1] = 0.01*Total_land_area[1]
UNITS: Hectares
Ag_land_limit[2] = 1000
UNITS: Hectares
Canal_adjustment[Villages] = IF(((TIME > 52*(2025-1965)) AND Agriculture.Proportion<0.1) AND Policies.Full_rewetting = 1) THEN 0 ELSE (IF(Depth_of_water_table < 0.4) THEN (IF (TIME > 52*15) THEN "1_m"/DT ELSE 0) ELSE 0)
UNITS: Meters / Week
canal_drain[Villages] = Extreme_Value_Test:_Canal_building* (IF (TIME >= 52*15) THEN "1_m_/_wk"+Canal_adjustment ELSE 0)

UNITS: Meters/Week
Climate_change = 1*1
UNITS: mm/Week
Deviation[Villages] = IF(Total_land_area < Error_check_1) THEN ABS(Total_land_area-Error_check_1)/Week ELSE 0
UNITS: Hectares/Week
ENSO = IF(TIME<= (2021-1965)*52) THEN ENSO_data ELSE ENSO_Model
UNITS: ENSO/ENSO
ENSO_adjusted_weekly_rainfall = IF(ENSO_Rainfall_model < 0) THEN 0 ELSE ENSO_Rainfall_model
UNITS: mm
ENSO_data = GRAPH(TIME)
UNITS: ENSO/ENSO
ENSO_Model = (-1.25*(SIN((TIME+0.25*26)/(52*5/(PI*2)))))
UNITS: ENSO/ENSO
ENSO_Rainfall_model = (Weekly_rainfall* (1- 0.25* (IF (ENSO*10 >= 7 AND ((TIME MOD 52) < 25.7) OR ((TIME MOD 52) > 47.57)) THEN 0 ELSE 1)* (IF (ENSO*10 <= -7 AND ((TIME MOD 52) < 25.7) OR ((TIME MOD 52) > 47.57)) THEN 0 ELSE 1)* (IF (ENSO*10 >= 7 AND ((TIME MOD 52) > 25.7) OR ((TIME MOD 52) < 47.57)) THEN 5 ELSE 1)* (IF (ENSO*10 <= -7 AND ((TIME MOD 52) > 25.7) OR ((TIME MOD 52) < 47.57)) THEN 0.5 ELSE 1) * ENSO)) +-10*TIME*(Climate_change-1)/5200
UNITS: mm
Error_check_1[Villages] = Built_village_extent+Forest_extent+Shrubs_or_degraded+Agriculture.Agricultural_land+Agriculture.Alternative_cultivation
UNITS: ha
Extreme_Value_Test:_Canal_building = 1
UNITS: Meters / Meters
Extreme_Value_Test:_Forestry = 1
UNITS: Hectares/Hectares
Extreme_Value_Test:_Peat_Layer = 1
UNITS: Meter / Meter
Fire_subsidence = 0.2
UNITS: Meters
Forestry[Villages] = Extreme_Value_Test:_Forestry* IF(TIME > 52*15 AND TIME < 52*50 AND Forest_extent > 0.1) THEN 0.02/52*INIT(Forest_extent) MOD Forest_extent ELSE 0
UNITS: Hectares
"Indicator_-_free_area"[1] = IF (Total_land_area[1]-Agriculture.Agricultural_land[1]>=0) THEN 1 ELSE 0
UNITS: Hectares / Hectares
"Indicator_-_free_area"[2] = IF (Total_land_area[2]-Agriculture.Agricultural_land[2]>=0) THEN 1 ELSE 0
UNITS: Hectares / Hectares
Inflow_to_water_table[Villages] = (ENSO_Rainfall_model/Rainfall_to_water_table_ratio-0.5)
UNITS: Meters
Land_constraints[1] = 10
UNITS: Hectares
Land_constraints[2] = 992
UNITS: Hectares
Land_per_person[Villages] = GRAPH(TIME)
UNITS: Hectares/People
Peat_layer[1] = IF(Peatland_layer_depth[1]*Extreme_Value_Test:_Peat_Layer-Subsided_peat_layer[1] > 0) THEN Peatland_layer_depth[1]-Subsided_peat_layer[1] ELSE 0
UNITS: Meter
Peat_layer[2] = IF(Peatland_layer_depth[2]*Extreme_Value_Test:_Peat_Layer-Subsided_peat_layer[2] > 0) THEN Peatland_layer_depth[2]-Subsided_peat_layer[2] ELSE 0

UNITS: Meter
Peat_swamp_vulnerability_to_fire[Villages] = (IF(Depth_of_water_table<= 0) THEN 0 ELSE 1)
UNITS: Meter / Meter
Peatland_layer_depth[1] = 10
UNITS: Meters
Peatland_layer_depth[2] = 5
UNITS: Meters
Rainfall = (140*SIN((TIME+0.25*26)*PI/26)+245)
UNITS: mm
Rainfall_to_water_table_ratio = 80
UNITS: mm/Meters
Rate_of_fire_subsidence[Villages] = Fire_subsidence *(Fire.Burnt_shrub_area+Deforestation)/Total_land_area
UNITS: Meter/Week
Rate_of_regeneration[Villages] = IF(Depth_of_water_table <=0) THEN .1/52 ELSE .01/52
UNITS: Hectares / Hectares
Rate_of_Subsidence[Villages] = (IF(Depth_of_water_table > 0) THEN 1 ELSE 0)* (IF(Peat_layer > 0) THEN 1 ELSE 0)* IF (((RoS_linear_variable*Weekly_water_table+RoS_constant)/Weeks_per_year)<=0) THEN 0 ELSE (IF (Peat_layer<Weekly_water_table) THEN ((RoS_linear_variable*Peat_layer+RoS_constant)/Weeks_per_year) ELSE ((RoS_linear_variable*Weekly_water_table+RoS_constant)/Weeks_per_year))
UNITS: Meter/Week
Ref_mode:_Kal_Ag_land = GRAPH(TIME)
UNITS: Hectares
Ref_mode:_Kal_Built_extent = GRAPH(TIME)
UNITS: Hectares
Ref_Mode:_Kal_degraded_land = GRAPH(TIME)
UNITS: Hectares
Ref_mode:_Kal_Forest_extent = GRAPH(TIME)
UNITS: Hectares
Ref_mode:_TN_Ag_land = GRAPH(TIME)
UNITS: Hectares
Ref_mode:_TN_Built_extent = GRAPH(TIME)
UNITS: Hectares
Ref_Mode:_TN_degraded_land = GRAPH(TIME)
UNITS: Hectares
Ref_mode:_TN_Forest_extent = GRAPH(TIME)
.810514), (5199, 4531.810514) {GF EXTRAPOLATED}
UNITS: Hectares
Rewetting_indicator[Villages] = IF(((TIME > 52*(2025-1965)) AND Agriculture.Proportion<Transition_threshold) AND Policies.Full_rewetting = 1) THEN 1 ELSE 0
UNITS: Metre / Metre
RoS_constant[Villages] = 1.24/100
UNITS: Meter / Year
RoS_linear_variable[Villages] = 0.0431
UNITS: Meter / Meter / Year
ST:_Transition_Threshold = 1*0.1/0.1*1
UNITS: Tonne / Tonne
Total_land_area[1] = 19482.69089
UNITS: Hectares
Total_land_area[2] = 3392.865705
UNITS: Hectares
Transition_threshold[Villages] = 0.1*ST:_Transition_Threshold
UNITS: Tonne / Tonne

Week = 1
UNITS: Week
Weekly_rainfall = Rainfall/(52/12)
UNITS: mm
Weekly_water_table[Villages] = Depth_of_water_table-Inflow_to_water_table
UNITS: Meter
Weeks_per_year = 52
UNITS: Weeks / Year

Policies:
Alternative_livelihood_policy[Villages] = 1
UNITS: IDR / IDR
Alternative_livelihood_policy_cap[1] = IF(TIME +"Weeks_/_Year"*Initial_year < Alternative_livelihood.Alternative_livelihood_program_commencement_year[1]*"Weeks_/_Year") THEN 0.1 ELSE (IF((Scenario=1 OR Scenario=2)) THEN 0.1 ELSE IF(Scenario=3 OR Scenario=4) THEN 1 ELSE 0)
UNITS: People / People
Alternative_livelihood_policy_cap[2] = IF(TIME +"Weeks_/_Year"*Initial_year < Alternative_livelihood.Alternative_livelihood_program_commencement_year[1]*"Weeks_/_Year") THEN 0.5 ELSE (IF(Scenario=1 OR Scenario=2) THEN 0.5 ELSE IF(Scenario=3 OR Scenario=4) THEN 1 ELSE 0)
UNITS: People / People
Full_rewetting = IF(Scenario=1 OR Scenario = 2) THEN 0 ELSE 1
UNITS: People/People
Initial_year = 1965
UNITS: Year
Permanent_flooding[Villages] = IF(Scenario=1) THEN 0 ELSE 1
UNITS: Meter / Meter
Scenario = 1
UNITS: Unit / Unit
"Weeks_/_Year" = 52
UNITS: Weeks/Year

Tests:
Kal_Hotspots = GRAPH(TIME)
UNITS: Hectares
TN_Hotspots = GRAPH(TIME)
UNITS: Hectares

Biochar:
BC_per_hectare[Villages](t) = BC_per_hectare[Villages](t - dt) + (Biochar_application[Villages] - Biochar_mineralisation[Villages]) * dt {NON-NEGATIVE}
INIT BC_per_hectare[Villages] = 0
UNITS: Tonnes / Hectare
Total_biochar[Villages](t) = Total_biochar[Villages](t - dt) + (BC_application[Villages]) * dt {NON-NEGATIVE}
INIT Total_biochar[Villages] = 0
UNITS: Tonnes
BC_application[Villages] = Agriculture.Agricultural_land[2]*Biochar_application[2] {UNIFLOW}
UNITS: Tonnes/Week
Biochar_application[Villages] = Biochar_cumulation_rate {UNIFLOW}
UNITS: Tonnes / Hectare/Weeks
Biochar_mineralisation[Villages] = BC_per_hectare*EXP(-10)/DT {UNIFLOW}
UNITS: Tonnes / Hectare/Weeks
Aggregate_Biochar_Yield_Effect[Villages] =

Biochar_application_rate/Targeted_application_multiplier*Biochar_yield_effect*Biochar_use*
(IF(BC_per_hectare>0) THEN 1 ELSE 0)*Hectare + (BC_per_hectare/Year-
Biochar_application_rate)* Yield_accumulation*Cumulative_biochar_effect*Biochar_use*Hectare
UNITS: Tonne/Tonne/Year
+(Total_biochar-Biochar_application_rate)*.0.1/10*Cumulative_effect
Biochar_application_rate[Villages] = 1
UNITS: Tonne/Hectare/Year
Biochar_cumulation_rate[Villages] = IF (TIME > 52*60) THEN
Biochar_application_rate/Weeks_per_year*Biochar_use ELSE 0
UNITS: Tonne / Hectare/Weeks
Biochar_use[Villages] = IF(ST:_Biochar_use = 1) THEN 1 ELSE 0
UNITS: Tonne / Tonne
Biochar_yield_effect[Villages] = 0.02*ST:_BC_yield_effect
UNITS: Tonne / Tonne
Cumulative_biochar_effect[Villages] = 1
UNITS: Tonne/Tonne
Hectare = 1
UNITS: Hectare
ST:_BC_yield_effect[Villages] = 1*1
UNITS: Tonne / Tonne
ST:_Biochar_use = 1*1
UNITS: Person / Person
Targeted_application_multiplier = 0.05
UNITS: Tonne
Weeks_per_year[Villages] = 52
UNITS: Week / Year
Year = 1
UNITS: Year
Yield_accumulation[Villages] = .01
UNITS: Hectare/Hectare/Tonne

Alternative_livelihood:
Alternative_Labourers[Villages](t) = Alternative_Labourers[Villages](t - dt) +
(New_alternatives[Villages] - Alternatives_leave[Villages]) * dt {NON-NEGATIVE}
INIT Alternative_Labourers[Villages] = 0
UNITS: People
Alternatives_leave[Villages] = IF (Alternative_Labourers <= 0) THEN 0 ELSE
IF(Expected_Alternative_Average_Income< Labour.Max_Income) THEN (Alternative_Labourers
MOD Labour.Mod_throughput)/Week ELSE 0 {UNIFLOW}
UNITS: People/Week
New_alternatives[Villages] = Alternative_livelihood_cap* Labour.New_Alternative {UNIFLOW}
UNITS: People/Week
"52wk_MA_Alternative_Income_per_worker"[Villages] =
MEAN(Alternative_livelihood_per_worker, DELAY(Alternative_livelihood_per_worker, 1),
DELAY(Alternative_livelihood_per_worker, 2), DELAY(Alternative_livelihood_per_worker, 3),
DELAY(Alternative_livelihood_per_worker, 4), DELAY(Alternative_livelihood_per_worker, 5),
DELAY(Alternative_livelihood_per_worker, 6), DELAY(Alternative_livelihood_per_worker, 7),
DELAY(Alternative_livelihood_per_worker, 8), DELAY(Alternative_livelihood_per_worker, 9),
DELAY(Alternative_livelihood_per_worker, 10), DELAY(Alternative_livelihood_per_worker, 11),
DELAY(Alternative_livelihood_per_worker, 12), DELAY(Alternative_livelihood_per_worker, 13),
DELAY(Alternative_livelihood_per_worker, 14), DELAY(Alternative_livelihood_per_worker, 15),
DELAY(Alternative_livelihood_per_worker, 16), DELAY(Alternative_livelihood_per_worker, 17),
DELAY(Alternative_livelihood_per_worker, 18), DELAY(Alternative_livelihood_per_worker, 19),
DELAY(Alternative_livelihood_per_worker, 20), DELAY(Alternative_livelihood_per_worker, 21),
DELAY(Alternative_livelihood_per_worker, 22), DELAY(Alternative_livelihood_per_worker, 23),

DELAY(Alternative_livelihood_per_worker, 24), DELAY(Alternative_livelihood_per_worker, 25),
DELAY(Alternative_livelihood_per_worker, 26), DELAY(Alternative_livelihood_per_worker, 27),
DELAY(Alternative_livelihood_per_worker, 28), DELAY(Alternative_livelihood_per_worker, 29),
DELAY(Alternative_livelihood_per_worker, 30), DELAY(Alternative_livelihood_per_worker, 31),
DELAY(Alternative_livelihood_per_worker, 32), DELAY(Alternative_livelihood_per_worker, 33),
DELAY(Alternative_livelihood_per_worker, 34), DELAY(Alternative_livelihood_per_worker, 35),
DELAY(Alternative_livelihood_per_worker, 36), DELAY(Alternative_livelihood_per_worker, 37),
DELAY(Alternative_livelihood_per_worker, 38), DELAY(Alternative_livelihood_per_worker, 39),
DELAY(Alternative_livelihood_per_worker, 40), DELAY(Alternative_livelihood_per_worker, 41),
DELAY(Alternative_livelihood_per_worker, 42), DELAY(Alternative_livelihood_per_worker, 43),
DELAY(Alternative_livelihood_per_worker, 44), DELAY(Alternative_livelihood_per_worker, 45),
DELAY(Alternative_livelihood_per_worker, 46), DELAY(Alternative_livelihood_per_worker, 47),
DELAY(Alternative_livelihood_per_worker, 48), DELAY(Alternative_livelihood_per_worker, 49),
DELAY(Alternative_livelihood_per_worker, 50), DELAY(Alternative_livelihood_per_worker, 51))
UNITS: IDR/People/Week

Alternative_livelihood_cap[Villages] = IF(Labour.Workforce <=0) THEN 0 ELSE IF(Alternative_Labourers/(Policies.Alternative_livelihood_policy_cap*Labour.Workforce) >= 1) THEN 0 ELSE 1
UNITS: People/People

Alternative_livelihood_per_worker[Villages] = Policies.Alternative_livelihood_policy* (IF(Fire.Air_pollution_index > 2000) THEN 0.75 ELSE IF(Fire.Air_pollution_index > 1000) THEN 0.9 ELSE 1)* (IF(TIME <=(2015-1965)*52) THEN 0 ELSE 1) * ((1+Biochar.Aggregate_Biochar_Yield_Effect*ST:_Biochar_Alternative_Livelihood))* Policies.Alternative_livelihood_policy + ((Initial_alternative_livelihood* (1+ Income_per_capita_growth/52) ^((TIME-40*52)/Week)))
UNITS: IDR/People/Week

Alternative_livelihood_program_commencement_year[Villages] = 2025
UNITS: Year

Alternative_total_income[Villages] = Alternative_livelihood_per_worker*Alternative_Labourers
UNITS: IDR/Week

Expected_Alternative_Average_Income[Villages] = IF(Policies.Alternative_livelihood_policy_cap*Labour.Workforce<=Alternative_Labourers) THEN 0 ELSE "52wk_MA_Alternative_Income_per_worker"
UNITS: IDR/People/Week

Income_per_capita_growth[Villages] = 0.0315
UNITS: Rate/Rate

Initial_alternative_livelihood[Villages] = 0.8*10^6 * ST:_Alt_livelihood *(IF(TIME > (Alternative_livelihood_program_commencement_year-1965)*"Weeks_/_Year") THEN 1 ELSE 0)
UNITS: IDR/People/Week

ST:_Alt_livelihood = 1*1
UNITS: IDR / IDR

ST:_Biochar_Alternative_Livelihood[Villages] = 1*1
UNITS: Year*IDR/People/Week

Week = 1
UNITS: Week

"Weeks_/_Year" = 52
UNITS: Weeks / Year

Fire:

Ag_Farmer_Ignition_Point_Frequency = ST:_Ignition_Point_Frequency*0.2
UNITS: IP / People / Week

Air_pollution_index[Villages] = Burnt_shrub_area[1]*Pollution_calibration
UNITS: µg/m^3/Week

Burn_area_:_Hotspot_ratio = 15.5
UNITS: Hectare / IP

Burnt_Forest_Area[Villages] = (Land_use.Forest_extent*Land_use.Peat_swamp_vulnerability_to_fire)*Prob_of_forest_burning*(IF(Land_use.ENSO_adjusted_weekly_rainfall<30) THEN 1 ELSE 0) / Week
UNITS: Hectares/Week
Burnt_shrub_area[1] = Hotspots[1] *Burn_area_:_Hotspot_ratio
UNITS: Hectares / Week
Burnt_shrub_area[2] = Hotspots[2] *Burn_area_:_Hotspot_ratio
UNITS: Hectares / Week
Expectation_of_PKY_becoming_capital = IF (52*(2019.8-1965) > TIME AND TIME > 52*(2019-1965)) THEN 1 ELSE 0
UNITS: People/People
General_population_Ignition_Point_Frequency = .045*ST:_Ignition_Point_Frequency
UNITS: IP / People / Week
Hotspots[1] = Potential_Hotspots[1]*vulnerability_of_degraded_peat_swamp_to_burning[1]/10^6*Land_use.Shrubs_or_degraded[1]
UNITS: IP / Week
Hotspots[2] = Potential_Hotspots[2]*vulnerability_of_degraded_peat_swamp_to_burning[2]/10^6*Land_use.Shrubs_or_degraded[2]*(1+Expectation_of_PKY_becoming_capital*17)
UNITS: IP / Week
Ignition_points[Villages] = (IF(Land_use.ENSO_adjusted_weekly_rainfall<30) THEN 1 ELSE 0)*(Agriculture.Farmers*Ag_Farmer_Ignition_Point_Frequency+ (.Population-Agriculture.Farmers)*General_population_Ignition_Point_Frequency)
UNITS: IP / Week
Pollution_calibration[Villages] = 70
UNITS: µg/m^3/hectare
Potential_Hotspots[1] = IF(Land_use.ENSO<0) THEN 0 ELSE 5*Ignition_points[1]*EXP(Land_use.ENSO)
UNITS: IP / Week
Potential_Hotspots[2] = IF(Land_use.ENSO<0) THEN 0 ELSE 3*Ignition_points[2]*EXP(Land_use.ENSO)
UNITS: IP / Week
Prob_of_forest_burning[Villages] = 1/100/52
UNITS: Hectares / Hectares
ST:_Ignition_Point_Frequency = 1*1
UNITS: IP / People / Week
Total_Burn_Area[Villages] = Burnt_shrub_area+Burnt_Forest_Area
UNITS: Hectares/Week
vulnerability_of_degraded_peat_swamp_to_burning[Villages] = (IF(Land_use.ENSO_adjusted_weekly_rainfall<30) THEN ABS((Land_use.ENSO_adjusted_weekly_rainfall - 25)/Vulnerability_ratio) ELSE 0) *(IF(Land_use.Weekly_water_table<0.041) THEN 0 ELSE 1)
UNITS: mm/mm/Hectare
Vulnerability_ratio[Villages] = 20
UNITS: Hectare*mm
Week = 1
UNITS: Week

GHG_emissions:
Net_accumulated_GHG_emissions_from_reduced_AGB[Villages](t) = Net_accumulated_GHG_emissions_from_reduced_AGB[Villages](t - dt) + (Mineralisation[Villages] - Regeneration[Villages]) * dt
INIT Net_accumulated_GHG_emissions_from_reduced_AGB[Villages] = 0
UNITS: Tonnes

```
Total_PS_accumulated_emissions[Villages](t) = Total_PS_accumulated_emissions[Villages](t - dt) + (Total_PS_emissions_rate[Villages]) * dt {NON-NEGATIVE}
   INIT Total_PS_accumulated_emissions[Villages] = 0
   UNITS: Tonnes
Mineralisation[Villages] = IF( (CO2e_sequestered_in_AGB-DELAY(CO2e_sequestered_in_AGB, 1))/Week < 0 )  THEN  -1/Week*(CO2e_sequestered_in_AGB-DELAY(CO2e_sequestered_in_AGB, 1))  ELSE 0 {UNIFLOW}
   UNITS: Tonnes/Week
Regeneration[Villages] = IF( (CO2e_sequestered_in_AGB-DELAY(CO2e_sequestered_in_AGB, 1))/Week > 0 )  THEN  (CO2e_sequestered_in_AGB-DELAY(CO2e_sequestered_in_AGB, 1))/Week ELSE 0 {UNIFLOW}
   UNITS: Tonnes/Week
Total_PS_emissions_rate[Villages] = Carbon_content_in_tonnes_per_hectare_of_1m_deep_peat*CO2e:C *(Land_use.Rate_of_Subsidence+Land_use.Rate_of_fire_subsidence)  *(IF(Land_use.Peat_layer>0) THEN 1 ELSE 0)  *(Degraded_land_vulnerable_to_fire) {UNIFLOW}
   UNITS: Tonnes/Week
"1m_depth" = 1
   UNITS: meter
AGB_GHG_flux[Villages] = Mineralisation-Regeneration-DELAY(Mineralisation-Regeneration, 1)
   UNITS: Tonnes/Week
Annualised_Emissions_per_capita[Villages] = IF(.Population = 0) THEN 0 ELSE Total_PS_emissions_rate/.Population*Weeks_per_year
   UNITS: Tonnes / Person / Year
Carbon_content_in_tonnes_per_hectare_of_1m_deep_peat[Villages] = Peat_C_content*Peatland_bulk_density  *"m^3/ha" / "1m_depth"
   UNITS: Tonnes / hectare / Meter
CO2e_sequestered_in_AGB[Villages] = Forest_AGB_Total_C+Shrubland_BGB_Total_C
   UNITS: Tonnes
CO2e:C = 44/12
   UNITS: Tonnes/Tonnes
Degraded_land_vulnerable_to_fire[Villages] = IF(Land_use.Peat_swamp_vulnerability_to_fire>0) THEN (Land_use.Shrubs_or_degraded+Agriculture.Agricultural_land) ELSE 0
   UNITS: Hectares
Forest_AGB[Villages] = 29.8*0.5
   UNITS: Tonnes / hectare
Forest_AGB_Total_C[Villages] = Land_use.Forest_extent*Forest_CO2e
   UNITS: Tonnes
Forest_CO2e[Villages] = Forest_AGB*CO2e:C
   UNITS: Tonnes/hectare
"m^3/ha"[Villages] = 10000
   UNITS: Cubic Metre / Hectare
Peat_C_content[Villages] = 0.56
   UNITS: Tonne / Tonne
Peatland_bulk_density[Villages] = .09
   UNITS: Tonne / Cubic Metre
Ref_Mode:_Hooijer_et_al_2006 = 220
   UNITS: tonne CO2e / yr / ha
Ref_Mode:_Hoscilo_2009 = 78.57
   UNITS: tonne CO2e / yr / ha
Shrubland_AGB[Villages] = 0.5*(0.5)+8.5*(0.2)
   UNITS: Tonnes/Hectare
Shrubland_BGB_Total_C[Villages] = Shrubland_CO2e*Land_use.Shrubs_or_degraded
   UNITS: Tonnes
Shrubland_CO2e[Villages] = Shrubland_AGB*CO2e:C
```

UNITS: Tonnes/Hectare
Total_accumulated_emissions[Villages] =
Total_PS_accumulated_emissions+Net_accumulated_GHG_emissions_from_reduced_AGB-Biochar.Total_biochar*CO2e:C
UNITS: Tonne
"Total_accumulated_emissions_in_tonnes_CO2e_/_ha_/_yr"[Villages] =
Total_accumulated_emissions/Land_use.Total_land_area/(TIME/Weeks_per_year)
UNITS: Tonne / hectare / year
Total_emissions_rate[Villages] = Total_PS_emissions_rate+AGB_GHG_flux
UNITS: Tonnes/Week
Week = 1
UNITS: Week
Weeks_per_year = 52
UNITS: Week / Year

{ The model has 411 (723) variables (array expansion in parens).
In root model and 10 additional modules with 0 sectors.
Stocks: 23 (46) Flows: 44 (88) Converters: 344 (589)
Constants: 82 (115) Equations: 306 (562) Graphicals: 20 (27)
}