APPENDIX A: code testing

The code testing is a check of appropriateness between what was wanted to be programmed and what was programmed. It was applied to all major submodels. SimPolilla© was programmed using CORMAS (CIRAD, France, http://cormas.cirad.fr) based on the VisualWorks programming environment (CincomSmalltalk, http://www.cincomsmalltalk.com). Appropriateness was performed using other programming languages such as R (http://www.r-project.org/). Each submodel was checked for appropriateness independently.

1. MORTALITY

a) Crude mortality

CORMAS code

```
    forceOfMortalityDispersalRelated:=0.060.
    forceOfMortalityNaturalEnemies:=(0.129 + 0.162) / 2.
    probaSurvival:= (0 - ((forceOfMortalityDispersalRelated + forceOfMortalityNaturalEnemies)*5) )
exp.
    Result: 0.357900594547609
```

R code

```
    forceOfMortalityDispersalRelated<-0.06
    forceOfMortalityNaturalEnemies<-(0.129 + 0.162) / 2
    probaSurvival<-exp(-((forceOfMortalityDispersalRelated + forceOfMortalityNaturalEnemies)*5))
    Result: 0.3579006
```

Equivalence confirmed.

b) Temperature dependent mortality

CORMAS code

```
    survivalRateEggTecia := (1/298.16) - (1/tempKelvinPromedio).
    survivalRateEggTecia := (bEgg * survivalRateEggTecia).
    survivalRateEggTecia := (survivalRateEggTecia) exp.
    survivalRateEggTecia := ((aEgg * tempKelvinPromedio) /298.16) *survivalRateEggTecia.
    survivalRateEggTecia := survivalRateEggTecia / ((((((1/dEgg) - (1/tempKelvinPromedio)) * cEgg)
/ R) exp) + 1) + (((((1/fEgg) - (1/tempKelvinPromedio)) * eEgg) / R) exp)).

    survivalRateEggTecia <=0 ifTrue:[survivalRateEggTecia:=0].

    survivalRateLarvaTecia := (1/298.16) - (1/tempKelvinPromedio).
    survivalRateLarvaTecia := (bLarva * survivalRateLarvaTecia).
    survivalRateLarvaTecia := (survivalRateLarvaTecia) exp.
    survivalRateLarvaTecia := ((aLarva * tempKelvinPromedio) /298.16) *survivalRateLarvaTecia.
    survivalRateLarvaTecia := survivalRateLarvaTecia / ((((((1/dLarva) - (1/tempKelvinPromedio)) *
cLarva) / R) exp) + 1) + (((((1/fLarva) - (1/tempKelvinPromedio)) * eLarva) / R) exp)).

    survivalRateLarvaTecia <=0 ifTrue:[survivalRateLarvaTecia:=0].

    survivalRatePupaTecia := (1/298.16) - (1/tempKelvinPromedio).
    survivalRatePupaTecia := (bPupa * survivalRatePupaTecia).
    survivalRatePupaTecia := (survivalRatePupaTecia) exp.
    survivalRatePupaTecia := ((aPupa * tempKelvinPromedio) /298.16) *survivalRatePupaTecia.
    survivalRatePupaTecia := survivalRatePupaTecia / ((((((1/dPupa) - (1/tempKelvinPromedio)) *
cPupa) / R) exp) + 1) + (((((1/fPupa) - (1/tempKelvinPromedio)) * ePupa) / R) exp)).

    survivalRatePupaTecia <=0 ifTrue:[survivalRatePupaTecia:=0].

    self survivalRateTecia: survivalRatePupaTecia * survivalRateLarvaTecia * survivalRateEggTecia.
    self survivalRateTecia
```

R code

```
    survivalRateEggTecia <- (1/298.16) - (1/tempKelvinPromedio)
    survivalRateEggTecia <- (bEgg * survivalRateEggTecia)
    survivalRateEggTecia <- exp(survivalRateEggTecia)
    survivalRateEggTecia <- ((aEgg * tempKelvinPromedio) /298.16) *survivalRateEggTecia
    survivalRateEggTecia <- survivalRateEggTecia / (((exp((((1/dEgg) - (1/tempKelvinPromedio)) *
cEgg) / R) ) + 1) + (exp((((1/fEgg) - (1/tempKelvinPromedio)) * eEgg) / R)))

    survivalRateLarvaTecia <- (1/298.16) - (1/tempKelvinPromedio)
    survivalRateLarvaTecia <- (bLarva * survivalRateLarvaTecia)
    survivalRateLarvaTecia <- exp(survivalRateLarvaTecia)
    survivalRateLarvaTecia <- ((aLarva * tempKelvinPromedio) /298.16) *survivalRateLarvaTecia
    survivalRateLarvaTecia <- survivalRateLarvaTecia / (((exp((((1/dLarva) - (1/tempKelvinPromedio))
* cLarva) / R) ) + 1) + (exp((((1/fLarva) - (1/tempKelvinPromedio)) * eLarva) / R)))

    survivalRatePupaTecia <- (1/298.16) - (1/tempKelvinPromedio)
    survivalRatePupaTecia <- (bPupa * survivalRatePupaTecia)
    survivalRatePupaTecia <- exp(survivalRatePupaTecia)
    survivalRatePupaTecia <- ((aPupa * tempKelvinPromedio) /298.16) *survivalRatePupaTecia
    survivalRatePupaTecia <- survivalRatePupaTecia / (((exp((((1/dPupa) - (1/tempKelvinPromedio)) *
cPupa) / R) ) + 1) + (exp((((1/fPupa) - (1/tempKelvinPromedio)) * ePupa) / R)))

    survivalRateTecia <- survivalRateEggTecia * survivalRateLarvaTecia * survivalRatePupaTecia
```

Equivalence checked using temperatures ranging from 0 to 45℃.

2. ADULT MOTH DISPERSAL

CORMAS code

```
    "Emigration rate among moth populations"
    a:=0.015.
    "Fraction of adult moth emigrating per generation as a function of insect density"
    "Carrying capacity K"
    K :=1000.
    ye := 0.5 / (1 + (0 - (self teciaAdults - (K / 2)) / 75) exp ).

    "Number of individuals in a process of dispersion per diffusion"
    numMigre := ye * self teciaAdults.

    "Cell caracteristics (500 * 500 m)"
    longCell := 500.
    surface := longCell * longCell.

    "Part of the population which can fligh from 200 to 249 meters"
    distance := (Cormas randomFrom: 200 to: 249).
    yd := ((0 - a) * distance) exp.
    "Probability of leaving the actual cell with the hypothésis of direct path to another cell"
    yeReal := (surface - ((longCell - (distance * 2))**2))/surface.
    "Part of the population which leave the given cell"
    yeReal200 := yd * yeReal.

    distance := (Cormas randomFrom: 150 to: 199).
    yd := ((0 - a) * distance) exp.
    yeReal := (surface - ((longCell - (distance * 2))**2))/surface.
    yeReal150 := yd * yeReal.

    distance := (Cormas randomFrom: 100 to: 149).
    yd := ((0 - a) * distance) exp.
    yeReal := (surface - ((longCell - (distance * 2))**2))/surface.
    yeReal100 := yd * yeReal.

    distance := (Cormas randomFrom: 75 to: 99).
    yd := ((0 - a) * distance) exp.
    yeReal := (surface - ((longCell - (distance * 2))**2))/surface.
    yeReal75 := yd * yeReal.

    distance := (Cormas randomFrom: 50 to: 74).
    yd := ((0 - a) * distance) exp.
    yeReal := (surface - ((longCell - (distance * 2))**2))/surface.
    yeReal50 := yd * yeReal.

    distance := (Cormas randomFrom: 25 to: 49).
    yd := ((0 - a) * distance) exp.
    yeReal := (surface - ((longCell - (distance * 2))**2))/surface.
    yeReal25 := yd * yeReal.

    distance := (Cormas randomFrom: 10 to: 24).
    yd := ((0 - a) * distance) exp.
    yeReal := (surface - ((longCell - (distance * 2))**2))/surface.
    yeReal10 := yd * yeReal.
    distance := (Cormas randomFrom: 5 to: 9).
    yd := ((0 - a) * distance) exp.
    yeReal := (surface - ((longCell - (distance * 2))**2))/surface.
    yeReal5 := yd * yeReal.

    distance := 1.
    yd := 1. "we assume that each PTM is moving in a 1 m² square"
    yeReal := (surface - ((longCell - (distance * 2))**2))/surface.
    yeReal1 := yd * yeReal.

    "Insect number leaving the given cell"
    dispersal := (yeReal200 + yeReal150 + yeReal100 + yeReal75 + yeReal50 + yeReal25 + yeReal10 +
yeReal5 + yeReal1) * numMigre.

    self bufferTecia: self bufferTecia - dispersal.
    self bufferTecia

    "Repartition of leaving insects in the neighborhood"
    self neighbourhood do: [:aCell | aCell bufferTecia: aCell bufferTecia + (dispersal / self
neighbourhood size)].
    self teciaAdults: self teciaAdults + self bufferTecia.
```

Equivalence checked constructing figures 1, 2 and 3 of the model description (using R).