carbon_model.rb

Rail trip carbon model

This model is used by Brighter Planet's carbon emission web service to estimate the **greenhouse gas emissions of passenger rail travel**.

Calculations

The final estimate is the result of the **calculations** detailed below. These calculations are performed in reverse order, starting with the last calculation listed and finishing with the emission calculation. Each calculation is named according to the `value` it returns.

Timeframe and date

The model estimates the emissions that occur during a particular `timeframe`. To do this it needs to know the trip's `date`. For example, if the `timeframe` is January 2010, a trip that occurred on January 11, 2010 will have emissions but a trip that occurred on Febraury 1, 2010 will not.

Methods

To accomodate varying client input, each calculation may have one or more **methods**. These are listed under each calculation in order from most to least preferred. Each method is named according to the `values` it requires ('default' methods do not require any values). Methods are ignored if any of

```
module BrighterPlanet
  module RailTrip
    module CarbonModel
      def self.included(base)
        base.decide :emission, :with =>
:characteristics do
```

the values they require are unvailable. Calculations are ignored if all of their methods are unavailable.

Standard compliance

Each method lists any established calculation standards with which it **complies**. When compliance with a standard is requested, all methods that do not comply with that standard are ignored. This means that any values a particular method requires will have been calculated using a compliant method or will be unavailable.

Collaboration

Contributions to this carbon model are actively encouraged and warmly welcomed. This library includes a comprehensive test suite to ensure that your changes do not cause regressions. All changes should include test coverage for new functionality. Please see sniff, our emitter testing framework, for more information.

Emission calculation

Returns the `emission` estimate ($kg\ CO_2e$).

From fuel and passengers

Complies: GHG Protocol Scope 3, ISO-140641, Climate Registry Protocol

```
committee :emission do
```

```
quorum 'from date, fuel, emission factors, and passengers', :needs => [:diesel_consumed, :diesel_emission_factor, :electricity_consumed, :electricity_emission_factor, :passengers, :date],
```

```
:complies => [:ghg_protocol_scope_3, :iso, :tcr] do |characteristics, timeframe|
```

```
date =
```

Checks whether the trip occurred during the `timeframe`

- Multiplies `diesel use` (l) by the `diesel emission factor` (kg / l) to give diesel emissions ($kg\ CO_2$)
- Multiplies `electricity use` (kWh) by the `electricity emission factor` ($kg\ CO_2e / kWh$) to give electricity emissions ($kg\ CO_2e$)
- Adds diesel and electricity emissions to give total emissions ($kg\ CO_2e$)
- Divides by passengers to give emissions per passenger ($kg\ CO_2e$)

If the trip did not occur during the `timeframe`, `emission` is zero

```
characteristics[:date].is_a?(Date) ?
characteristics[:date] :
Date.parse(characteristics[:date].to_s)

              if timeframe.include? date

(characteristics[:diesel_consumed] *
characteristics[:diesel_emission_factor] +
characteristics[:electricity_consumed] *
characteristics[:electricity_emission_factor]) /
characteristics[:passengers]
              else

                0
              end
            end
          end

          committee :diesel_emission_factor do

            quorum 'default',

              :complies => [:ghg_protocol_scope_3,
:iso, :tcr] do
```

Diesel emission factor calculation

Returns the `diesel emission factor` ($kg\ CO_2e / l$).

Default diesel emission factor

Complies: GHG Protocol Scope 3, ISO 14064-1, Climate Registry Protocol

Looks up Distillate Fuel Oil No. 2's `co2 emission factor` (kg / l).

Electricity emission factor calculation

Returns the `electricity emission factor` ($kg\ CO_2e / kWh$).

Default electricity emission factor

Complies: GHG Protocol Scope 3, ISO 14064-1, Climate Registry Protocol

Looks up the U.S. Average `electricity emission factor` ($kg\ CO_2e / l$).

Looks up the U.S. Average grid region `loss factor`.

Divides the `electricity emission factor` by (1 – `loss factor`) to account for transmission and distribution losses.

Diesel consumed calculation

Returns the `diesel use` (l).

```
              diesel = Fuel.find_by_name
"Distillate Fuel Oil No. 2"
              diesel.co2_emission_factor
            end
          end



          committee :electricity_emission_factor
do



            quorum 'default',

              :complies => [:ghg_protocol_scope_3,
:iso, :tcr] do

                subregion =
EgridSubregion.find_by_abbreviation "US"

                region = subregion.egrid_region

                emission_factor =
subregion.electricity_emission_factor / (1 -
region.loss_factor)
                emission_factor
            end
          end



          committee :diesel_consumed do
```

From distance and diesel intensity

Complies: GHG Protocol Scope 3, ISO-140641, Climate Registry Protocol

Multiplies `distance` (km) by `diesel intensity` (l / km) to give l.

```
            quorum 'from distance and diesel
intensity', :needs => [:distance,
:diesel_intensity],

              :complies => [:ghg_protocol_scope_3,
:iso, :tcr] do |characteristics|

                characteristics[:distance] *
characteristics[:diesel_intensity]
            end
          end
```

Electricity consumed calculation

Returns the `electricity use` (kWh).

```
          committee :electricity_consumed do
```

From distance and electricity intensity

Complies: GHG Protocol Scope 3, ISO-140641, Climate Registry Protocol

Multiplies `distance` (km) by `electricity intensity` (kWh / km) to give kWh.

```
            quorum 'from distance and electricity
intensity', :needs => [:distance,
:electricity_intensity],

              :complies => [:ghg_protocol_scope_3,
:iso, :tcr] do |characteristics|

                characteristics[:distance] *
characteristics[:electricity_intensity]
            end
          end
```

Distance calculation

Returns the `distance` traveled (*km*).

Distance from distance estimate

Complies: GHG Protocol Scope 3, ISO-140641, Climate Registry Protocol

Uses the `distance estimate` (*km*).

Distance from duration and speed

Complies: GHG Protocol Scope 3, ISO-140641, Climate Registry Protocol

Multiplies the `duration` (*seconds*) (converted to hours) by the `speed` (*km / hour*) to give *km*.

Distance from rail class

Looks up the rail class `distance`.

```
          committee :distance do



            quorum 'from distance estimate',
:needs => :distance_estimate,

              :complies => [:ghg_protocol_scope_3,
:iso, :tcr] do |characteristics|

characteristics[:distance_estimate]
            end



            quorum 'from duration and speed',
:needs => [:duration, :speed],

              :complies => [:ghg_protocol_scope_3,
:iso, :tcr] do |characteristics|

                characteristics[:duration] /
3600.0 * characteristics[:speed]
            end



            quorum 'from rail class', :needs =>
:rail_class do |characteristics|

characteristics[:rail_class].distance
```

Distance estimate calculation

Returns the trip's `distance estimate` (*km*)

Distance estimate from client input

Complies: All

Uses the client-input `distance estimate` (*km*).

Duration calculation

Returns the trip's `duration` (*seconds*).

Duration from client input

Complies: All

Uses the client-input `duration` (*seconds*).

Diesel intensity calculation

Returns the `diesel intensity` (l / km).

Diesel intensity from rail class

Complies: GHG Protocol Scope 3, ISO-140641, Climate Registry Protocol

```
            end
          end

          committee :diesel_intensity do

            quorum 'from rail class', :needs =>
:rail_class,

              :complies => [:ghg_protocol_scope_3,
:iso, :tcr] do |characteristics|
```

Looks up the rail class `diesel intensity`.

```
characteristics[:rail_class].diesel_intensity
            end
          end
```

Electricity intensity calculation

Returns the `electricity intensity` (kWh / km).

```
          committee :electricity_intensity do
```

Electricity intensity from rail class

```
            quorum 'from rail class', :needs =>
:rail_class,
```

Complies: GHG Protocol Scope 3, ISO-140641, Climate Registry Protocol

```
              :complies => [:ghg_protocol_scope_3,
:iso, :tcr] do |characteristics|
```

Looks up the rail class `electricity intensity`.

```
characteristics[:rail_class].electricity_intensity
            end
          end
```

Speed calculation

Returns the average `speed` ($km / hour$).

```
          committee :speed do
```

Speed from rail class

```
            quorum 'from rail class', :needs =>
:rail_class,
```

Complies: GHG Protocol Scope 3, ISO-140641, Climate Registry Protocol

```
              :complies => [:ghg_protocol_scope_3,
:iso, :tcr] do |characteristics|
```

Looks up the rail class `speed`.

Passengers calculation

Returns the total number of `passengers`.

Passengers from rail class

Complies: GHG Protocol Scope 3, ISO-140641, Climate Registry Protocol

Looks up the rail class `passengers`.

Rail class calculation

Returns the rail class. This is the type of rail the trip used.

Rail class from client input

Complies: All

Uses the client-input rail class.

```
              characteristics[:rail_class].speed
            end
          end


          committee :passengers do


            quorum 'from rail class', :needs =>
:rail_class,

              :complies => [:ghg_protocol_scope_3,
:iso, :tcr]  do |characteristics|

characteristics[:rail_class].passengers
            end
          end


          committee :rail_class do
```

Default rail class

Complies: GHG Protocol Scope 3, ISO 14064-1, Climate Registry Protocol

Uses an artificial rail class representing the U.S. average.

```
            quorum 'default',

              :complies => [:ghg_protocol_scope_3,
:iso, :tcr] do

                RailClass.find_by_name "US
average"
            end
          end
```

Date calculation

Returns the `date` on which the trip occurred.

Date from client input

Complies: All

Uses the client-input `date`.

```
          committee :date do
```

Date from timeframe

Complies: GHG Protocol Scope 3, ISO-14064-1, Climate Registry Protocol

Assumes the trip occurred on the first day of the `timeframe`.

```
            quorum 'from timeframe',

              :complies => [:ghg_protocol_scope_3,
:iso, :tcr] do |characteristics, timeframe|

                timeframe.from
            end
          end
```

Timeframe calculation

Returns the `timeframe`. This is the period during which to calculate emissions.

Timeframe from client input

Complies: All

Uses the client-input `timeframe`.

Default timeframe

Complies: All

Uses the current calendar year.

```
        end
      end
    end
  end
end
```