New Jersey

Carbon Dividend Calculator

Product Guide

Jonathan Lu – Project Leader

Agata Foryciarz

Changyan Wang

Joseph Abbate

User guide

Overview

A **carbon fee and dividend policy** places a fee on carbon-dioxide emitting-fuels such as coal, natural gas, and petroleum. Ultimately it raises the price of things like gasoline, home heating, and fossil fuel-based electricity. This internalizes the hidden costs of air pollution and climate change, and encourages consumers and businesses to switch to cleaner and more efficient forms of energy. A majority of the fees are returned to households as dividends in order to help them cope with increased energy costs.

The **NJ Carbon Dividend Calculator** is a web application that gives NJ residents an estimate for the net cost or benefit they will receive from a proposed carbon fee and dividend policy under consideration. It also allows the user to see how changes in their energy use behavior (e.g. turning off the lights more often) could change their estimated benefit or cost. Our goal is to provide this service with a **transparent, nonpartisan, secure,** and **simple** webapp.

Calculating your Carbon Fee and Dividend

The website can be accessed at https://njcdc.herokuapp.com/.

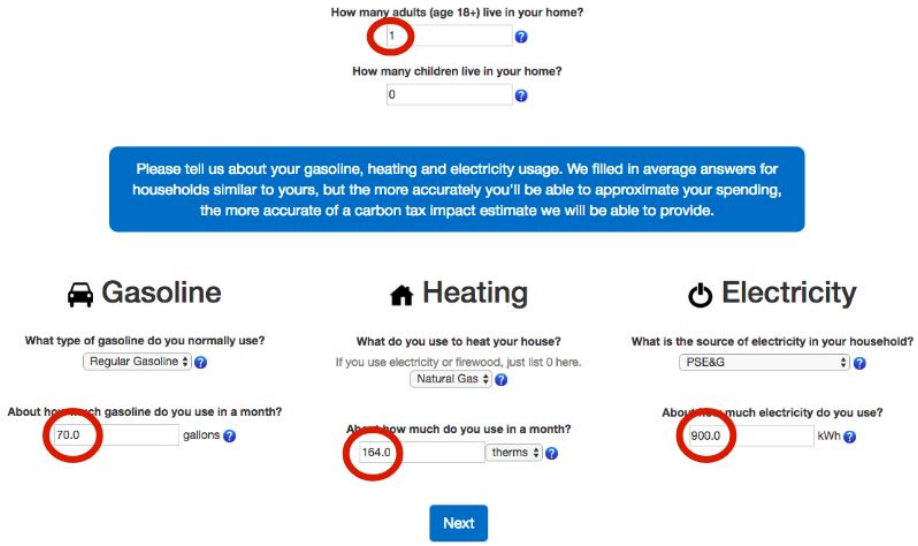
Let's walk through an example to illustrate how the calculator is used. Suppose I am a single man living in South Jersey. I drive a truck, and don't keep track of my electricity or heating spending, but assume it's close to average. I worry that the tax will make keeping my tank full prohibitively expensive.

Household Information

First, I enter information about the number of adults and children in my household. This will determine how much dividend I will receive.

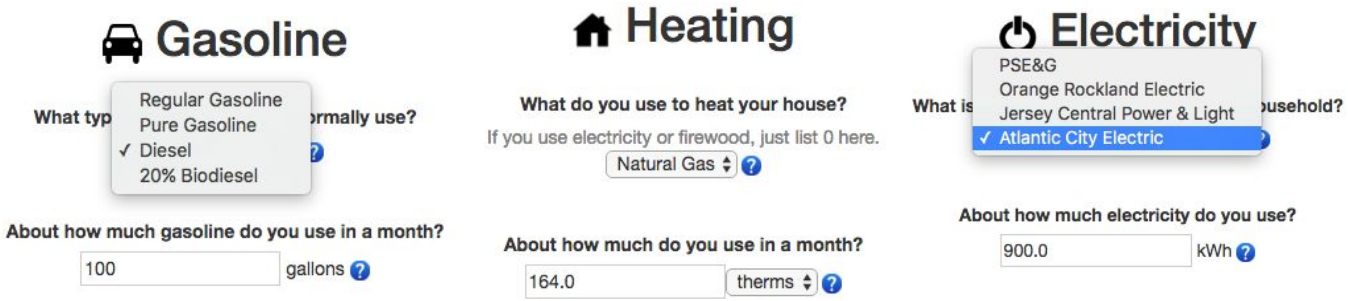
Default Inputs

The values have been pre-filled with average inputs. I can use those if I am not sure about my spending - so I will leave the heating and electricity values the same. However, the types of inputs will have to be changed.



Energy Sources

We will select Diesel for gasoline source and increase monthly gasoline usage to 100. We will keep heating and electricity usage the same, but will change the electric company to Atlantic City Electric.

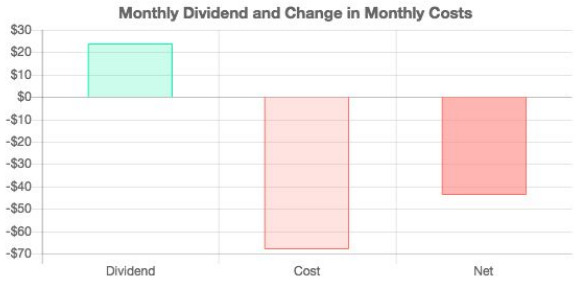


Viewing Results

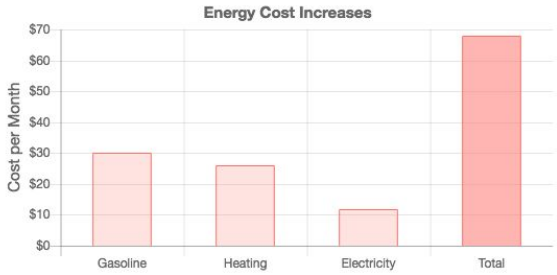
After filling out inputs and clicking “Next”, I will be taken to the Results page. The page will display the amount of money that the tax will cost or earn me, as well as a detailed breakdown of how the use of different energy sources is contributing to the tax.

Our calculations show that the current policy will **cost** you **$44** per month.

You'd pay **$68** more in energy costs per month, but would receive a **$24** tax credit per month.



Based on your inputs, **44%** of your tax would come from **Gasoline** spending, **38%** would come from your **Heat** spending and **18%** would come from your **Electricity** spending.



It looks like the policy will make me lose money! I see that Gasoline and Heating are the main contributors to my tax. I am offered to see the ways in which I can decrease my tax:

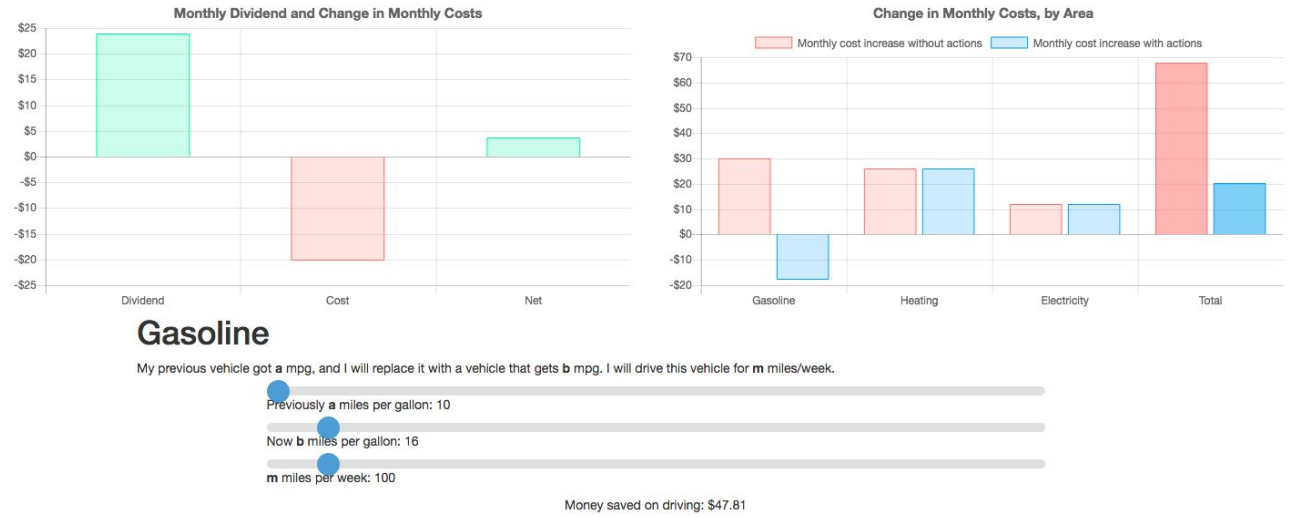
Wondering what you can do to decrease your tax? Click below to see how changing your energy, gasoline and heating usage habits could help lower your tax.

Actions

Actions

On the actions page, I am shown three options: Gasoline, Heating, and Electricity to change my energy usage. I examine how behavior changes can impact my bottom line using the sliders - the graphs update in real time! Right now they are set to the original values I had from the previous page.

My current truck drives 10 miles per gallon. I do not want to drive any less, but I have been considering buying a new car. After setting the parameters for current car and number of miles, and playing with the new car parameters, it looks like I will come out positive if my new car drives at least 16 miles per gallon.



Developer Guide

We used Python 3.6.4 with Django 2.0.3.

Installation

Clone the github repo at https://github.com/lujonathanh/njcdc. Install virtualenv and install Python3 and Django 2.0. Installing a database engine is optional.

Importantly, you *must* modify settings.py for local development. See the README for detailed setup instructions, including settings and database setup. To populate your database with data for NJ zip codes, run the populate_database script in the Django Python shell (`python manage.py shell < data/populate_database.py`).

Back End

Host

We host our site on Heroku. Our workflow is standard for Python Heroku projects. If you wish to clone our website on a different URL, update ALLOWED_HOSTS in Django settings. If you wish to contribute to the njcdc.herokuapp.com website, you should fork our Github and submit a pull request. If approved, we will push the changes to the Heroku live server.

Data Cleaning

The data is from Berkeley's CoolClimate. You can request the data here. We cleaned the data by removing data from other states and filling in missing values. For details, see data/data_cleaning_info.txt.

Django

The main input model is the UserProfile, defined in njcdc/calc/models.py. This takes in user household and energy information and calculates their net benefit and cost under the policy. The model defines each field and sanitizes each input using the validators. It imports default values from njcdc/calc/parameters.py. Once the values are filled in by the user, it calculates the co2 footprint from each source of energy using generic functions defined out-of-class (to enable testing without creation of the costs). It calculates the benefit and finally the net cost.

The main input form is InputForm, defined in njcdc/calc/forms.py. This simply allows users to fill in the parameters of the UserProfile. Importantly, several fields are excluded from the user: the details of the policy (fee, rebate_portion, period) and the units of certain items (gasoline_unit, elec_unit). The policy details are excluded because these should only be set based on the policy under consideration. Gasoline_unit and elec_unit are excluded because gallons and kWh are the two fundamental units we shall use. (even if one wanted to input in terms of miles traveled, ultimately this would be converted into the gallon unit of gasoline).

The three main views are input(), results(), and actions(), all defined in njcdc/calc/views.py. input() provides the energy information input; results displays the net cost and cost breakdown calculations; and actions demonstrates how various kinds of actions will change one's costs. Input creates a UserProfile, performs the calculation, then stores the values in a cookie and deletes the object. This facilitates data passing without any database storage; thus privacy is not an issue.

Calculating the Fee, Rebate, and Defaults

To calculate the fee for the user, we multiplied their energy usage times the amount of CO_2 emitted by that energy source, and then multiplied by the fee per unit of CO_2. We used data from the EPA, power companies, and others, and links to these data are included in our models file.
The rebate is proportional to the size of the household. We estimated the amount of rebate using EPA data on annual New Jersey emissions, the size of the fee, the percent rebated to households, and the total New Jersey population.

To estimate the default values for New Jersey (for prepopulating our fields), we used data from the EPA, NJ government, EIA, and CoolClimate (links and formulas included in models).

Adapting the Website

To update the policy design, update the relevant parameters in the Django parameters.py under the section “POLICY PARAMETERS” (FEE, REBATE_PORTION, and CHILD_MULTIPLIER). This will automatically update the example calculations in the Policy page as well.

To update the defaults, update the relevant one under section INPUT DEFAULTS.

To update the state:

1. Update the state emissions and population parameters under section STATE-SPECIFIC PARAMETERS.
2. Update the gasoline, electric, and heating parameters (below)
3. Update the database (under zipcode_data) with other state data.

To update the gasoline, electric, and heating parameters:

- If updating a value, just fill in the new value. Include citations for consistency.
- If ADDING a choice (e.g. a gasoline or heating option):
    1. Add it as a (internal_name, external_name) tuple to the respective list of choices, e.g. GASOLINE_CHOICES
    2. If the old unit (e.g. gallon) doesn’t apply to this new choice:
        a. Add a new unit choice, e.g. GASOLINE_UNIT_CHOICES
        b. Make the unit function return the unit as a possible unit for the new option: get_possible_gasoline_units
    3. Add a constant that gives the new option its ton CO2 per unit, e.g. e10_ton_co2_per_gallon
    4. Update the conversion function, e.g. get_gasoline_co2_conversion to get this new constant when performing the calculation.

Front End

(Relevant Code: njcdc/calc/templates/*)

Bootstrap

We use Bootstrap to generate responsive web design, in order to make our website usable on various devices. The graphics display side by side on large screen, in a single column on smaller devices. We also used Bootstrap's CSS library to adapt to aesthetic suggestions from our users, including the color scheme, font size and spacing.

We moved our custom bootstrap-based CSS settings to the calc/templates/calc/base.html file, which we import in each HTML file.

(Relevant Code: njcdc/calc/templates/results.html, njcdc/calc/templates/actions.html)

Interactive Graphs

The interactive graphs were implemented with JavaScript and JS libraries: jQuery and ChartJS. On the Actions page, we wanted our graphics to change dynamically as users moved the actions sliders. We also used JavaScript to make the graphics on the action page "stick" to the top of the page as users scroll through actions.