Theories of change

(and dplyr magic)

January 29, 2020

PMAP 8521: Program Evaluation for Public Service
Andrew Young School of Policy Studies
Spring 2020

Fill out your reading report on iCollege!

Plan for today

Manipulating data with `dplyr`

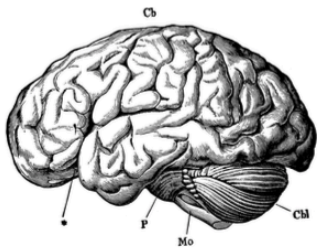
Program theories

Logic models & results chains

Manipulating data with dplyr

The tidyverse

The tidyverse



Human language

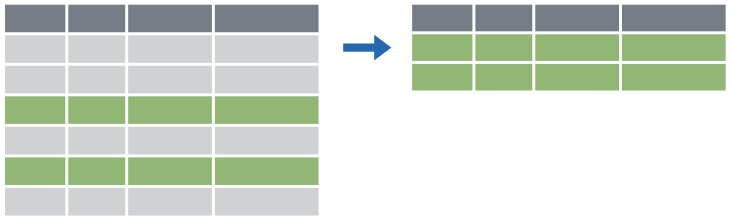
tidyverse

base R

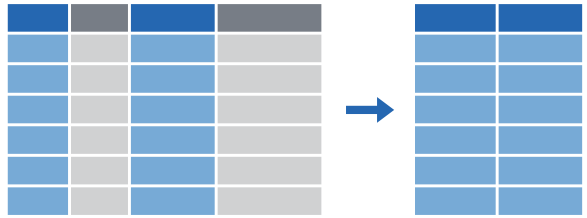
C++

Machine language

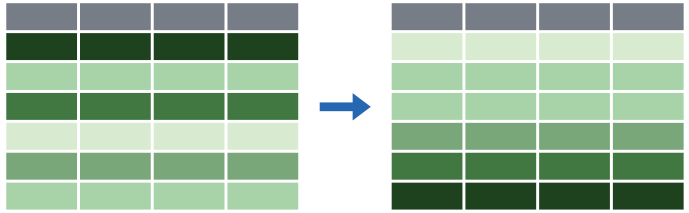
Most important dplyr verbs



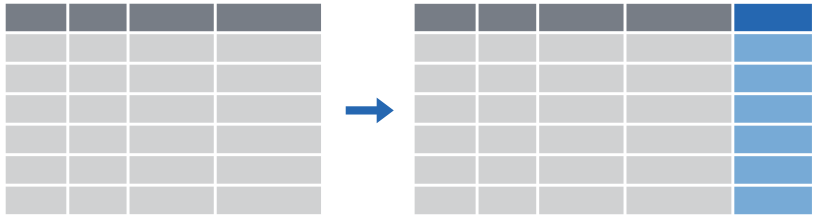
Extract rows/cases with `filter()`



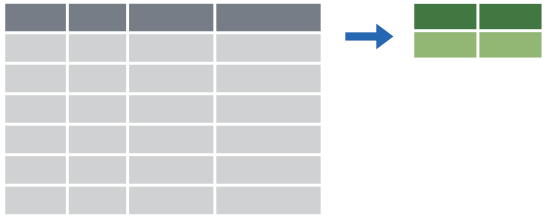
Extract columns/variables with `select()`



Arrange/sort rows with `arrange()`



Make new columns/variables with `mutate()`



Make group summaries with `group_by() %>% summarize()`

filter()

Extract rows that meet some sort of test

```
filter(.data, ...)
```

Data frame to transform

One or more tests

(filter returns each row for which the test is TRUE)

filter()

Extract rows that meet some sort of test

```
filter(gapminder, country == "Denmark")
```

country	continent	year	...
Afghanistan	Asia	1952	...
Afghanistan	Asia	1957	...
...	...	...	...
Czech Republic	Europe	2007	...
Denmark	Europe	1952	...
Denmark	Europe	1957	...
Denmark	...	...	...

country	continent	year	...
Denmark	Europe	1952	...
Denmark	Europe	1957	...
Denmark	Europe	1962	...
Denmark	Europe	1967	...
Denmark	Europe	1972	...
Denmark	Europe	1977	...
...	...	...	...

filter()

```
filter(gapminder, country == "Denmark")
```

One = sets an argument
(returns nothing)

Two == tests if equal
(returns TRUE or FALSE)

Logical tests

Test	Meaning
`x < y`	Less than
`x > y`	Greater than
`x == y`	Equal to
`x <= y`	Less than or equal to
`x >= y`	Greater than or equal to
`x != y`	Not equal to
`x %in% y`	In (group membership)
`is.na(x)`	Is missing
`!is.na(x)`	Is not missing

Your turn (#1)

Use `filter()` and logical tests to show...

1. The data for Canada
2. All data for countries in Oceania
3. Rows where the life expectancy is greater than 82

Your turn (#1)

Use `filter()` and logical tests to show...

1. The data for Canada
2. All data for countries in Oceania
3. Rows where the life expectancy is greater than 82

04:00

```
filter(gapminder, country == "Canada")
```

```
filter(gapminder, continent == "Oceania")
```

```
filter(gapminder, lifeExp > 82)
```

Common mistakes

Using = instead of ==

```
filter(gapminder, country = "Canada")
```

```
filter(gapminder, country == "Canada")
```

Quote use

```
filter(gapminder, country == Canada)
```

```
filter(gapminder, country == "Canada")
```

`filter()` with multiple conditions

Extract rows that meet *every* test

```
filter(gapminder, country == "Denmark", year > 2000)
```

country	continent	year	...
Afghanistan	Asia	1952	...
Afghanistan	Asia	1957	...
...	...	...	...
Czech Republic	Europe	2007	...
Denmark	Europe	1952	...
Denmark	...	...	...
Denmark	Europe	2002	...

country	continent	year	...
Denmark	Europe	2002	...
Denmark	Europe	2007	...

Boolean operators

Operator	Meaning
a & b	and
a | b	or
!a	not

`filter()` with multiple conditions

Extract rows that meet *every* test

```
filter(gapminder, country == "Denmark" & year > 2000)
```

country	continent	year	...
Afghanistan	Asia	1952	...
Afghanistan	Asia	1957	...
...	...	...	...
Czech Republic	Europe	2007	...
Denmark	Europe	1952	...
Denmark	...	...	...
Denmark	Europe	2002	...

country	continent	year	...
Denmark	Europe	2002	...
Denmark	Europe	2007	...

Your turn (#2)

Use `filter()` and Boolean logical tests to show...

1. Canada before 1970
2. Countries where life expectancy in 2007 is below 50
3. Countries where life expectancy in 2007 is below 50 and are not in Africa

Your turn (#2)

Use `filter()` and Boolean logical tests to show...

1. Canada before 1970
2. Countries where life expectancy in 2007 is below 50
3. Countries where life expectancy in 2007 is below 50 and are not in Africa

04:00

```
filter(gapminder, country == "Canada",
       year < 1970)
```

```
filter(gapminder, year == 2007, lifeExp < 50)
```

```
filter(gapminder, year == 2007, lifeExp < 50,
       continent != "Africa")
```

Common mistakes

Collapsing multiple tests into one

```
filter(gapminder, 1960 < year < 1980)
```

```
filter(gapminder, 1960 < year, year < 1980)
```

Stringing together many tests when you could use %in%

```
filter(gapminder, country == "Mexico" | country == "Canada" |
         country == "United States")
```

```
filter(gapminder, country %in% c("Mexico", "Canada",
                                 "United States"))
```

Common syntax

Every dplyr verb function follow the same pattern

First argument is a data frame; returns a data frame

```
<VERB>(.data, ...)
```

dplyr function/verb

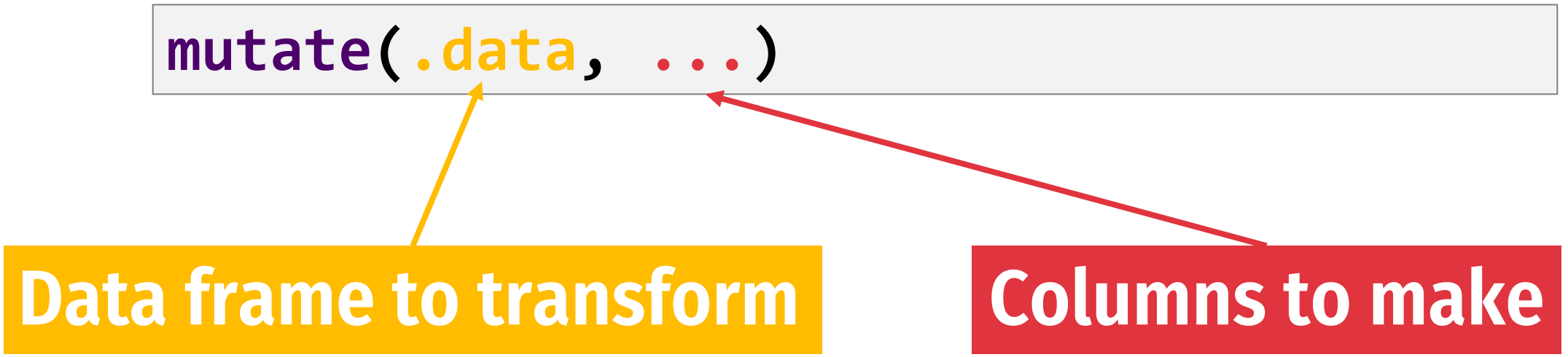
Stuff the verb does

Data frame to transform

mutate()

Create new columns

```
mutate(.data, ...)
```



mutate()

Create new columns

```
mutate(gapminder, gdp = gdpPercap * pop)
```

country	continent	year	...
Afghanistan	Asia	1952	...
Afghanistan	Asia	1957	...
Afghanistan	Asia	1962	...
Afghanistan	Asia	1967	...
Afghanistan	Asia	1972	...
Afghanistan	Asia	1977	...
Afghanistan	Asia	...	...

country	continent	year	...	gdp
Afghanistan	Asia	1952	...	6567086330
Afghanistan	Asia	1957	...	7585448670
Afghanistan	Asia	1962	...	8758855797
Afghanistan	Asia	1967	...	9648014150
Afghanistan	Asia	1972	...	9678553274
Afghanistan	Asia	1977	...	11697659231
Afghanistan	Asia	...	...	...

mutate()

Create new columns

```
mutate(gapminder, gdp = gdpPercap * pop,
                  pop_mill = round(pop / 1000000))
```

country	continent	year	...
Afghanistan	Asia	1952	...
Afghanistan	Asia	1957	...
Afghanistan	Asia	1962	...
Afghanistan	Asia	1967	...
Afghanistan	Asia	1972	...
Afghanistan	Asia	1977	...
Afghanistan	Asia	...	...

country	continent	year	...	gdp	pop_mill
Afghanistan	Asia	1952	...	6567086330	8
Afghanistan	Asia	1957	...	7585448670	9
Afghanistan	Asia	1962	...	8758855797	10
Afghanistan	Asia	1967	...	9648014150	12
Afghanistan	Asia	1972	...	9678553274	13
Afghanistan	Asia	1977	...	11697659231	15
Afghanistan	Asia	...	...	...	...

ifelse()

Do conditional tests within mutate()

```
ifelse(<TEST>, <VALUE IF TRUE>, <VALUE IF FALSE>)
```

```
mutate(gapminder,
       after_1960 = ifelse(year > 1960, TRUE, FALSE))
```

```
mutate(gapminder,
       after_1960 =
         ifelse(year > 1960, "After 1960", "Before 1960")
```

Your turn (#3)

Use `mutate()` to …

1. Add an `africa` column that is TRUE if the country is on the African continent
2. Add a column for logged GDP per capita
3. Add an `africa_asia` column that says “Africa or Asia” if the country is in Africa or Asia, and “Not Africa or Asia” if it’s not

Your turn (#3)

Use `mutate()` to …

1. Add an `africa` column that is TRUE if the country is on the African continent
2. Add a column for logged GDP per capita
3. Add an `africa_asia` column that says “Africa or Asia” if the country is in Africa or Asia, and “Not Africa or Asia” if it’s not

05:00

```
mutate(gapminder, africa = continent == "Africa")
```

```
mutate(gapminder, log_gdpPercap = log(gdpPercap))
```

```
mutate(gapminder,
       after_1960 = ifelse(year > 1960,
                           "After 1960",
                           "Before 1960"))
```

What if you have multiple verbs?

Make a dataset for just 2002; calculate log GDP per capita

Solution 1: Intermediate variables

```
gapminder_2002 <- filter(gapminder, year == 2002)

gapminder_2002_logged <- mutate(gapminder_2002,
                                log_gdpPercap =
                                  log(gdpPercap))
```

What if you have multiple verbs?

Make a dataset for just 2002; calculate log GDP per capita

Solution 2: Nested functions

```
filter(mutate(gapminder_2002,
              log_gdpPercap = log(gdpPercap)),
       year == 2002)
```

What if you have multiple verbs?

Make a dataset for just 2002; calculate log GDP per capita

Solution 3: Pipes!

The %>% (pipe) takes object on the left and passes it as the first argument of the function on the right

```
gapminder %>% filter(_______, country == "Canada")
```

What if you have multiple verbs?

These do the same thing!

```
filter(gapminder, country == "Canada")
```

```
gapminder %>% filter(country == "Canada")
```

What if you have multiple verbs?

Make a dataset for just 2002; calculate log GDP per capita

Solution 3: Pipes!

```
gapminder %>%
  filter(year == 2002) %>%
  mutate(log_gdpPercap = log(gdpPercap))
```

%>%

```
leave_house(get_dressed(get_out_of_bed(wake_up(me, time =
"8:00"), side = "correct"), pants = TRUE, shirt = TRUE),
car = TRUE, bike = FALSE)
```

```
me %>%
  wake_up(time = "8:00") %>%
  get_out_of_bed(side = "correct") %>%
  get_dressed(pants = TRUE, shirt = TRUE) %>%
  leave_house(car = TRUE, bike = FALSE)
```

summarize()

Compute table of summaries

```
gapminder %>% summarize(mean_life = mean(lifeExp))
```

country	continent	year	lifeExp	...
Afghanistan	Asia	1952	28.801	...
Afghanistan	Asia	1957	30.332	...
Afghanistan	Asia	1962	31.997	...
Afghanistan	Asia	1967	34.020	...
Afghanistan	Asia	1972	36.088	...
Afghanistan	Asia	...	...	...

mean_life
59.47444

summarize()

Compute table of summaries

```
gapminder %>% summarize(mean_life = mean(lifeExp),
                        min_life = min(lifeExp))
```

country	continent	year	lifeExp	...
Afghanistan	Asia	1952	28.801	...
Afghanistan	Asia	1957	30.332	...
Afghanistan	Asia	1962	31.997	...
Afghanistan	Asia	1967	34.020	...
Afghanistan	Asia	1972	36.088	...
Afghanistan	Asia	...	...	...

mean_life	min_life
59.47444	23.599

Your turn (#4)

Use `summarize()` to calculate...

1. The first (minimum) year in the dataset
2. The last (maximum) year in the dataset
3. The number of rows in the dataset (use the cheatsheet)
4. The number of distinct countries in the dataset (use the cheatsheet)

Your turn (#4)

Use `summarize()` to calculate...

1. The first (minimum) year in the dataset
2. The last (maximum) year in the dataset
3. The number of rows in the dataset (use the cheatsheet)
4. The number of distinct countries in the dataset (use the cheatsheet)

04:00

```
gapminder %>%
  summarize(first = min(year),
            last = max(year),
            num_rows = n(),
            num_unique = n_distinct(country))
```

```
# A tibble: 1 x 4
  first  last num_rows num_unique
  <int> <int>    <int>      <int>
1  1952  2007     1704        142
```

Your turn (#5)

Use `filter()` and `summarize()` to calculate the (1) the number of unique countries and (2) the median life expectancy on the African continent in 2007

Your turn (#5)

Use `filter()` and `summarize()` to calculate the (1) the number of unique countries and (2) the median life expectancy on the African continent in 2007

04:00

```
gapminder %>%
  filter(continent == "Africa", year == 2007) %>%
  summarise(n_countries = n_distinct(country),
            med_le = median(lifeExp))
```

```
# A tibble: 1 x 2
  n_countries med_le
        <int>  <dbl>
1          52   52.9
```

group_by()

Put rows into groups based on values in a column

```
gapminder %>% group_by(continent)
```

Nothing happens by itself!

Powerful when combined with `summarize()`

group_by()

```
gapminder %>%
  group_by(continent) %>%
  summarize(n_countries = n_distinct(country))
```

continent	n_countries
Africa	52
Americas	25
Asia	33
Europe	30
Oceania	2

group_by() %>% summarize()

city	particle_size	amount
New York	Large	23
New York	Small	14
London	Large	22
London	Small	16
Beijing	Large	121
Beijing	Small	56

mean	sum	n
42	252	6

```
pollution %>%
  summarize(mean = mean(amount), sum = sum(amount), n = n())
```

group_by() %>% summarize()

city	particle_size	amount
New York	Large	23
New York	Small	14
London	Large	22
London	Small	16
Beijing	Large	121
Beijing	Small	56

mean	sum	n
18.5	37	2

mean	sum	n
19.0	38	2

mean	sum	n
88.5	177	2

city	mean	sum	n
New York	18.5	37	2
London	19.0	38	2
Beijing	88.5	177	2

```
pollution %>%
  group_by(city) %>%
  summarize(mean = mean(amount), sum = sum(amount), n = n())
```

group_by() %>% summarize()

city	particle_size	amount
New York	Large	23
New York	Small	14
London	Large	22
London	Small	16
Beijing	Large	121
Beijing	Small	56

mean	sum	n
55.33	166	3

mean	sum	n
28.67	86	2

particle_size	mean	sum	n
Large	55.33	166	3
Small	28.67	86	3

```
pollution %>%
  group_by(particle_size) %>%
  summarize(mean = mean(amount), sum = sum(amount), n = n())
```

Your turn (#6)

Find the minimum, maximum, and median life expectancy for each continent

Find the minimum, maximum, and median life expectancy for each continent in 2007 only

Your turn (#6)

Find the minimum, maximum, and median life expectancy for each continent

Find the minimum, maximum, and median life expectancy for each continent in 2007 only

05:00

```
gapminder %>%
  group_by(continent) %>%
  summarize(min_le = min(lifeExp),
            max_le = max(lifeExp),
            med_le = median(lifeExp))
```

```
gapminder %>%
  filter(year == 2007) %>%
  group_by(continent) %>%
  summarize(min_le = min(lifeExp),
            max_le = max(lifeExp),
            med_le = median(lifeExp))
```

Program theories

Elements of a program

Inputs

Things that go into a project; money, people, time, etc.

Activities

Actions that convert inputs to outputs; things that you do

Outputs

Tangible goods and services produced by activities; you have control over these

Outcomes

What happens when the target population uses the outputs; you don’t have control over these

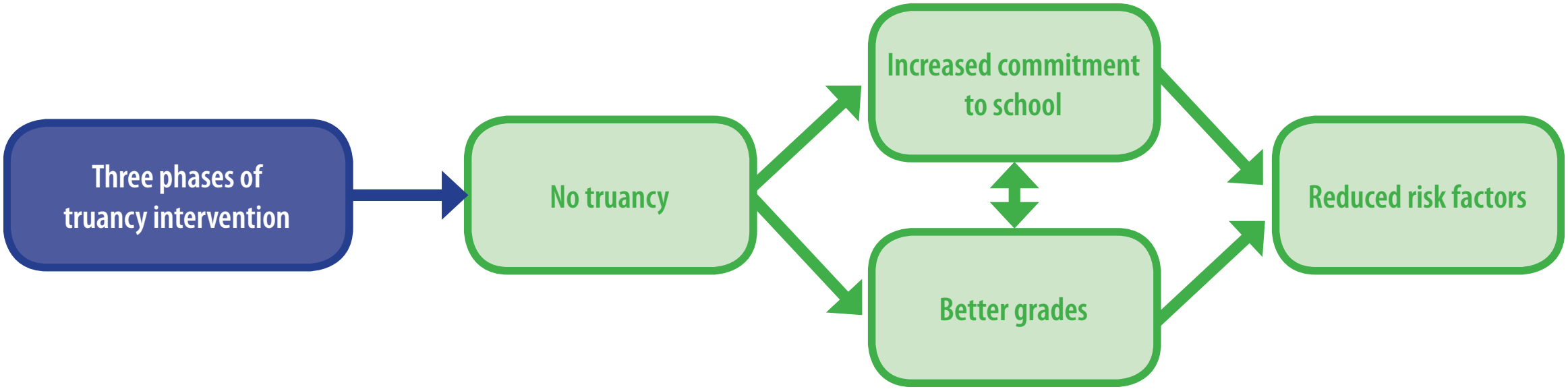
Program theory

How and why an intervention causes change

A sequence of events that connects inputs to activities to outputs to outcomes

Impact theory

Causes (activities) linked to effects (outcomes)



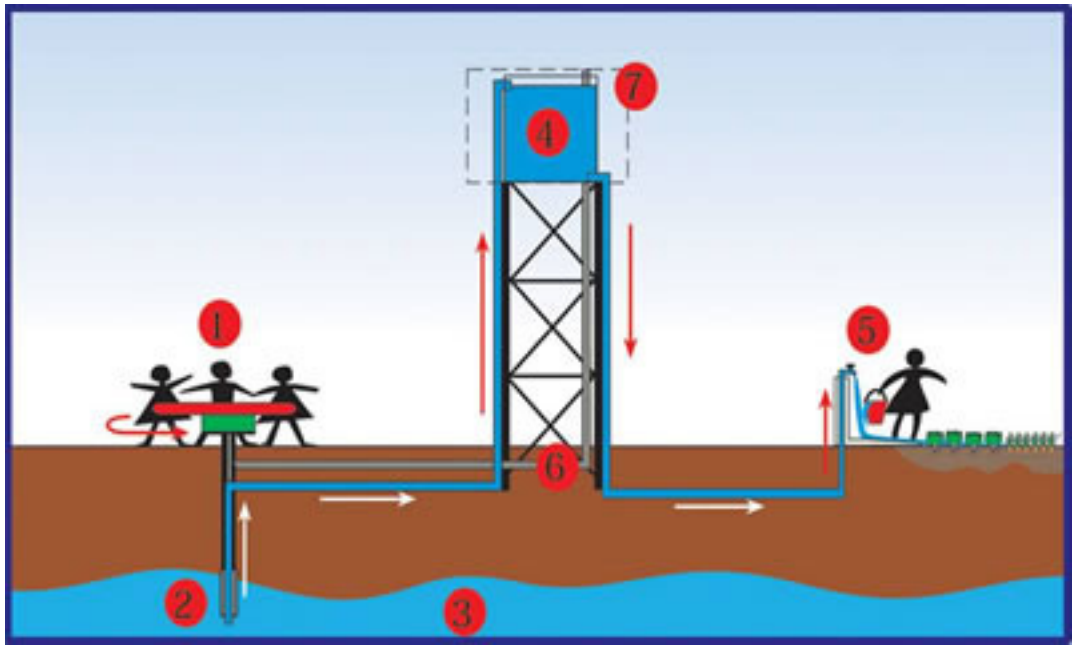
One Laptop Per Child (OLPC)

One Laptop Per Child (OLPC)

OLPC may have undercut even the XO-1's strong points by overselling them. "The utopianism set unrealistic expectations around what the laptops should be able to accomplish," says Morgan Ames, a Berkeley researcher who's currently writing a book about OLPC. That included Negroponte's laptop-tossing demonstrations. "When you're talking about a laptop that kids are using surrounded by concrete floors and cobblestone streets — there was a ton of breakage that really blindsided projects, because they expected these laptops to be a lot more indestructible."

"THE UTOPIANISM SET UNREALISTIC EXPECTATIONS AROUND WHAT THE LAPTOPS SHOULD BE ABLE TO ACCOMPLISH."

Playpump



Why theorize?

Should all social programs be rooted in explicit theory?

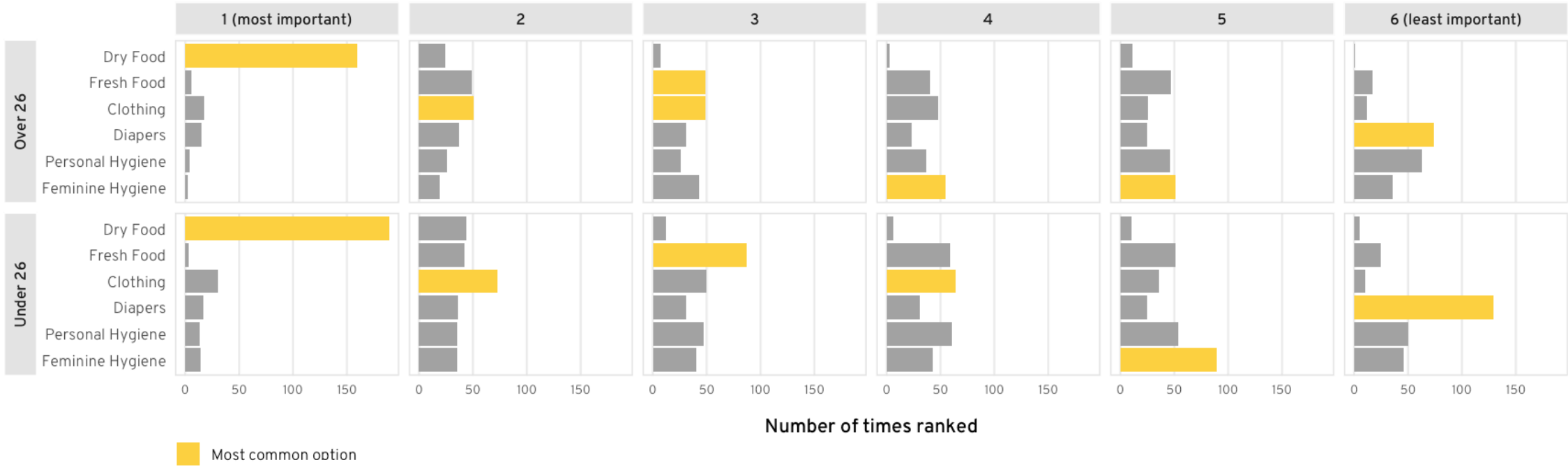
Articulated theory

Implicit theory

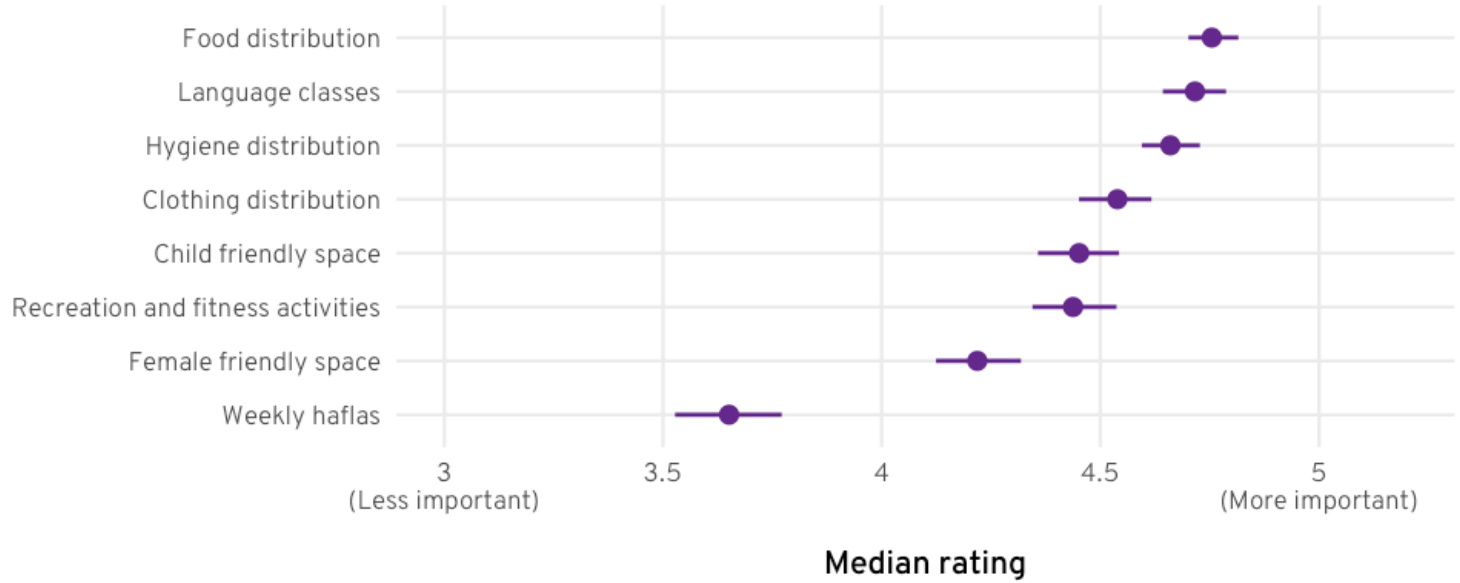

Mens Shoes
UK 9
men's Socks
Mens Dress Shoes


American Fork, Utah
Southern US Border
Greece
Kurdistan (Northern Iraq)
Jordan & Lebanon
Bangladesh
Somalia
Lifting Hands Around the World

Distribution preferences by age



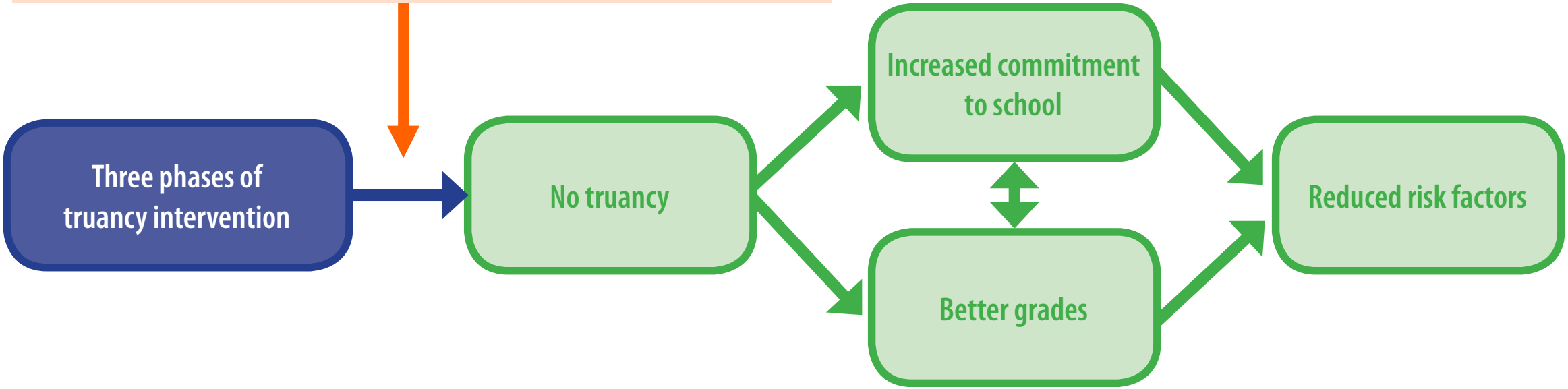
Median program importance, overall



Error bars show 95% highest-density credible interval

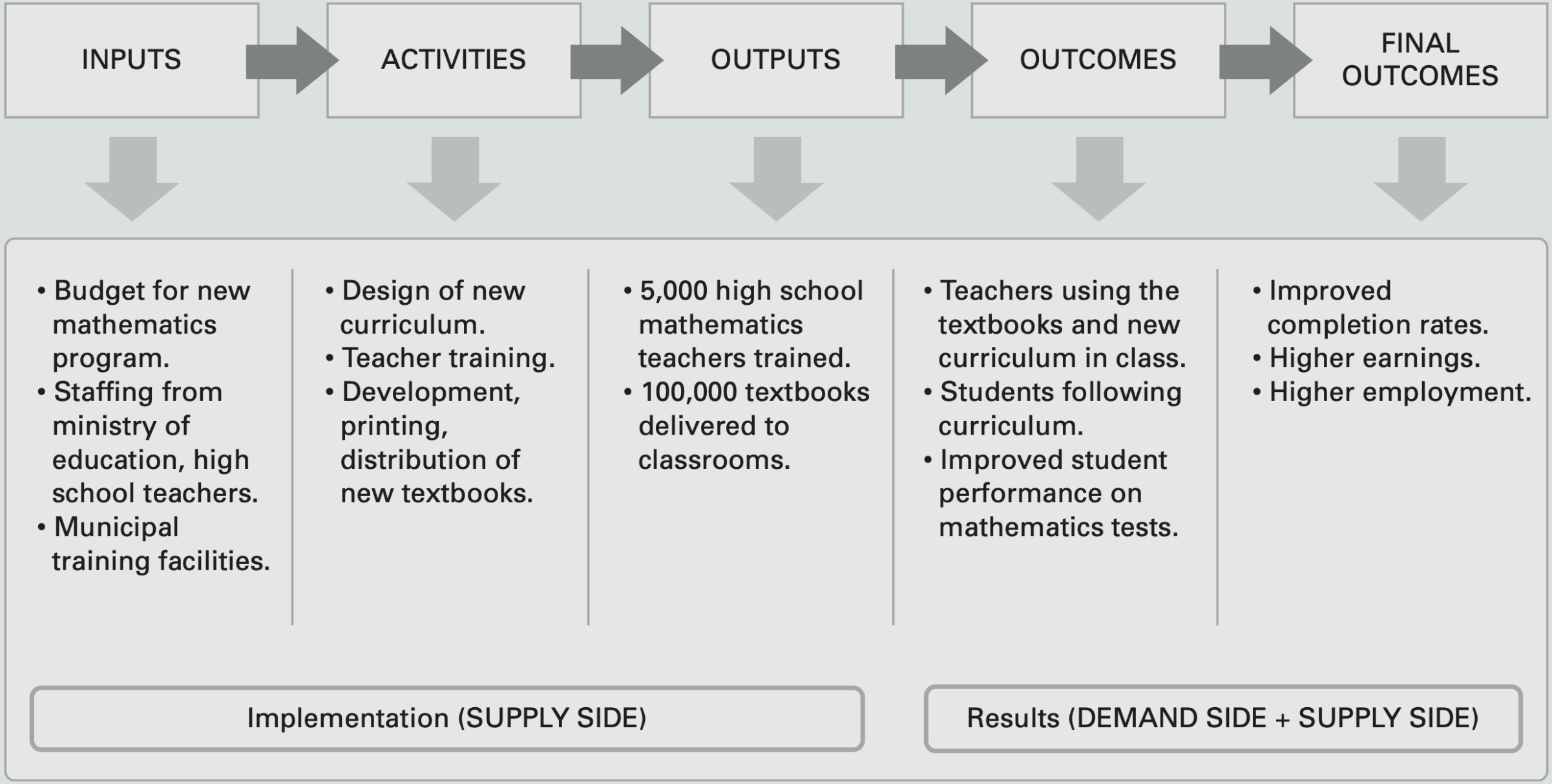
Impact theory

Ensure that the theory linking activities to the outcomes is sound!



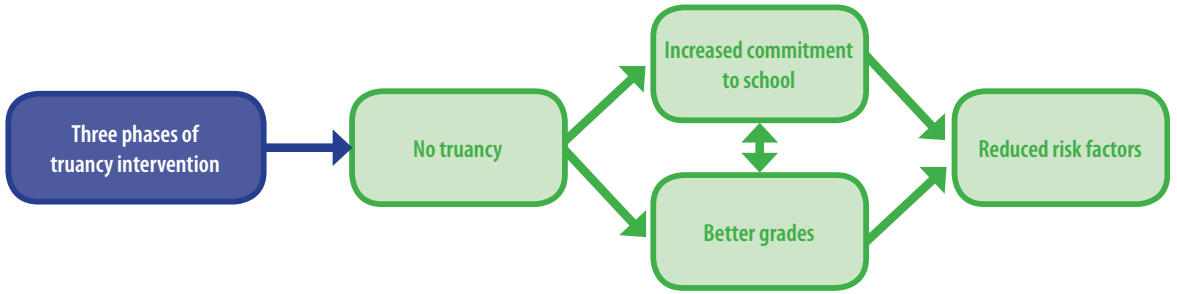
Logic models & results chains

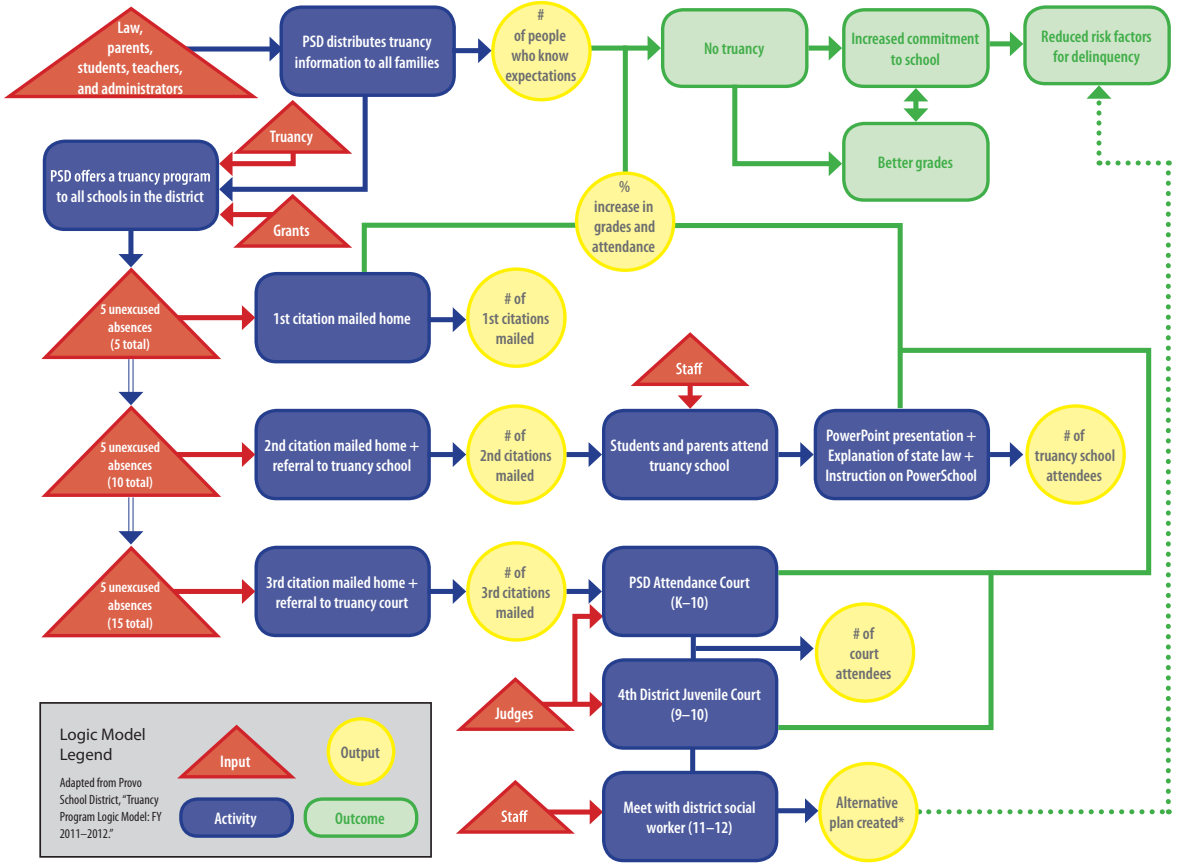
Figure B2.3.1 A Results Chain for the High School Mathematics Curriculum Reform



* Because 11th and 12th graders who receive 3rd citations are generally unable to graduate from high school, district social workers no longer attempt to increase their commitment to school. As such, any outcomes that occur as a result of the alternative plans made for these students (work study programs, career development assistance, etc.) are only tangentially related to the outcomes of the truancy program itself. The system for creating alternative plans is an entirely separate program with its own logic model, goals, and outcomes.

Impact theory vs. logic model





* Because 11th and 12th graders who receive 3rd citations are generally unable to graduate from high school, district social workers no longer attempt to increase their commitment to school. As such, any outcomes that occur as a result of the alternative plans made for these students (work study programs, career development assistance, etc.) are only tangentially related to the outcomes of the truancy program itself. The system for creating alternative plans is an entirely separate program with its own logic model, goals, and outcomes.

MPA/MPP at GSU

Master of Public Policy

Preparing students for roles as effective citizens and workers in the public sphere.

About | Curriculum | Admissions | MPA vs. MPP | Current Students

The Master of Public Policy (MPP) is an interdisciplinary degree program designed to prepare students for work in the analysis, development, and evaluation of public policies. In all levels of government and on a global scale, public needs and limited resources require public policy choices that are at once economically efficient, socially and technically effective, and politically responsive. Such choices confront policymakers in a broad range of critical issues, including health, education, economic development, public finance, social policy, nonprofit policy, and disaster policy.

Decision-makers often lack the knowledge and skills needed to interpret the full social, political, economic, and technical dimensions of the policy issues they face. In response, state and local governments, businesses, and federal agencies have turned to trained policy analysts for assistance in assessing policy options and in evaluating public programs. The same is true for nonprofit agencies, such as hospitals, schools, emergency preparedness and relief agencies, and regional planning organizations.

Master of Public Administration

A flexible program for working professionals and full-time scholars.

The mission of the Master of Public Administration (MPA) program is to prepare students to become leaders in public service careers as executives, managers, analysts, and policy specialists in government and nonprofit organizations.

Your own logic models