

Рис. 1.

Общие сведения

Скорость сообщения (скорость доставки) — это средняя скорость перемещения груза или пассажира из начального пункта отправления в конечный пункт назначения. Ключевое отличие этого показателя в том, что он учитывает все затраты времени: не только движение по маршруту, но и плановые и внеплановые остановки, простои, а также время на логистические операции — погрузку, разгрузку, сортировку, маневровые работы.

Математически скорость сообщения (V_c) выражается формулой:

$$V_c = L_m / (t_{дв} + t_{оп}),$$

где:

- L_m — длина маршрута;
- $t_{дв}$ — время движения;
- $t_{оп}$ — время остановок на промежуточных и конечных пунктах.

В транспортной логистике выделяют три взаимосвязанных показателя:

Вид скорости	Что учитывает
Техническая	Только время движения по маршруту, без простоев (остановки на перекрёстках, переездах). Определяется конструктивными возможностями транспорта и состоянием путей.
Эксплуатационная	Время движения плюс технологические остановки (погрузка, разгрузка, смена бригад, заправка).
Скорость сообщения	Все затраты времени: движение, плановые и внеплановые остановки, логистические операции. Это показатель реальной скорости доставки для клиента.

Практическое значение

Этот показатель критически важен для оценки качества транспортного обслуживания и планирования расписаний. Он напрямую влияет на удовлетворённость клиентов, так как отражает реальное время, которое они проведут в пути. Анализ скорости сообщения помогает выявлять «узкие места» в логистической цепи и оптимизировать работу всей транспортной системы.

Производительность транспортной системы — это показатель, который отражает объём транспортной работы (перевезённых грузов или пассажиров), выполненный за единицу времени. Он позволяет оценить эффективность использования инфраструктуры, подвижного состава и организации перевозок.

Для транспортной сети

Для сети в целом часто используют показатель суточной производительности — то есть объём работы, который может быть выполнен за сутки.

Формула для расчёта суточной производительности городской транспортной сети:

$$P_c = 2 \cdot L_m \cdot K_m \cdot N_n \cdot K_c \cdot T_{nv}$$

где:

- L_m — длина транспортной сети (км);
- K_m — коэффициент многополосности;
- N_n — пропускная способность одной полосы (авт./ч), рассчитываемая как $D \cdot V$,

где D — плотность движения (авт./км), V — скорость движения (км/ч);

- K_c — коэффициент снижения пропускной способности из-за неравномерности загрузки (обычно принимается равным 0,7);
- T_n — коэффициент приведения пиковой загрузки к суточной (принимается равным 10).

Для транспортных средств

Производительность отдельного транспортного средства характеризуется количеством перевезённых грузов (в тоннах) или выполненной работой (в тонно-километрах) за единицу времени.

Необходимо определить факторы, влияющие на производительность.

Энергоэффективность транспортной системы — это способность использовать меньше энергии для выполнения той же транспортной работы (перевозки пассажиров или грузов). Это не просто экономия топлива, а комплексное повышение рациональности энергопотребления на всех этапах: от добычи и доставки энергоресурсов до движения транспорта и управления потоками.

Практическая значимость

- **Экономическая выгода.** Снижение энергопотребления напрямую ведёт к уменьшению эксплуатационных расходов.
- **Экологическая устойчивость.** Меньшее потребление невозобновляемых ресурсов и снижение вредных выбросов.
- **Оптимизация инфраструктуры.** Повышение эффективности использования путей, станций и развязок.

Основные источники потерь энергии

- Сопротивление движению: аэродинамическое и гидродинамическое сопротивление, сопротивление качению и трение колёс о поверхность пути, сопротивление в механизмах, электромагнитные потери.
- Неэффективное управление: частые циклы «старт-стоп», задержки, пробки.
- Избыточная масса подвижного состава и инфраструктуры.
- Потери при торможении: энергия, которая могла бы быть возвращена при торможении, рассеивается в виде тепла.

Ключевые направления повышения энергоэффективности

Технологические решения

- **Электрификация.** Например электровозы имеют КПД до 95%, тогда как у тепловозов — всего 30–35%. Это делает электрический транспорт значительно более эффективным.
- **Рекуперативное торможение.** Технология, при которой кинетическая энергия преобразуется в электрическую и возвращается в сеть или используется для подзарядки аккумуляторов. Яркий пример — Московское центральное кольцо, где возвращается около 50% энергии.
- **Снижение массы транспортных средств** за счёт применения лёгких и прочных материалов (полимерные композиты, алюминий).
- **Оптимизация способов движения** с точки зрения сопротивления движению.

Интеллектуальные системы управления

- Автоматическое управление скоростью с учётом дорожных условий и потока транспорта.
- Энергооптимальные графики движения, прогнозные расписания.
- Мониторинг технического состояния ТС для своевременного устранения неисправностей, которые повышают расход энергии или топлива.

Условные циклы движения транспортных средств — это стандартизированные последовательности режимов движения, которые имитируют реальные условия эксплуатации и используются для испытаний транспортных средств, оценки расхода топлива или энергии, выбросов загрязняющих веществ, моделирования работы силовой установки и других характеристик.

Виды условных циклов движения

- 1.По зависимости от времени или расстояния:
 - Зависимые от времени — сжатая версия фактического времени испытания на дороге.
 - Зависимые от расстояния — фактическая копия испытательной дороги.
- 2.По характеру изменения скорости:
 - Переходные циклы — включают множество изменений скорости, типичны для вождения по бездорожью.
 - Режимные циклы — включают длительные периоды при постоянных скоростях.

Особенности реализации компьютерной модели.

Передача данных в рабочую область

Передать данные в рабочую область SCILAB можно с помощью блока *To workspace* из группы *Регистрирующие устройства*. Добавим этот блок в модель, построенную на предыдущем занятии.

На основной вход этого блока подадим выходной сигнал системы, а на верхний (синхронизирующий) вход — сигнал с часов (генератора тактов событий) (см. рис. 1).

Прежде, чем запускать моделирование, нужно задать параметры для нового блока — имя переменной и размер массива данных. Выберем имя переменной — **out**, а размер массива данных вычислим, разделив время моделирования (50 с) на период синхриимпульсов, которые приходят с часов (0,01 с). Таким образом, нужно установить такие параметры:

Size of buffer: **5000**

Scilab variable: **out**

Выполним моделирование и после его окончания выведем переменную **OUT_x** в командном окне SCILAB:

```
--> out  
out =
```

values: [4999x1 constant]

time: [4999x1 constant]

Согласно этой информации переменная **out** представляет собой структуру с двумя полями, каждое из которых — это массив из 4999 элементов. Поле **time** содержит отсчёты времени, а поле

values — соответствующие им значения измеряемого сигнала.

Для того чтобы построить график, можно использовать встроенную функцию **plot**.

Пусть в переменной **t** записан массив значений времени, а в переменной **y** — массив значений выхода

системы. Эти массивы должны быть одинакового размера. В простейшем случае функция **plot** вызывается так:

```
--> plot( t, y )
```

Построенный график можно сохранить в виде графического файла (меню *Файл – Экспортировать*) или скопировать в буфер обмена (меню *Файл – Копировать в буфер обмена*).

Функции **plot** можно передавать в качестве аргументов не целые массивы, а их части.

Например, по команде

```
--> plot( x(1:20), y(11:30) )
```

строится график, на котором по оси абсцисс откладываются первые 20 значений элементов массива **x**, а по оси ординат – 20 значений из массива **y** с номерами от 11 до 30 (включительно). Двоеточие означает «все строки» или «все столбцы». Например, по команде

```
--> plot( x(:,1), x(:,2) )
```

строится график зависимости между первым и вторым столбцами двумерного массива (матрицы) **x**. Здесь двоеточие вместо первого индекса обозначает «все строки». Если вызвать команду **plot** повторно с другими аргументами:

```
--> plot( t1, y1 )
```

новый график будет построен в том же окне, что и предыдущий.

Для создания нового окна для графика в SCILAB используется команда

```
--> h = scf()
```

В переменную **h** записывается ссылка (указатель) на объект –графическое окно. С помощью этой ссылки окном можно управлять, например, изменять цвет линий, размер шрифта и т.п.

После выполнения этой команды на экран выводятся все свойства графического окна, к которым можно обращаться через ссылку **h**. Начало списка выглядит так:

Handle of type "Figure" with properties:

=====

children: "Axes"

figure_position = [625,23]

figure_size = [618,573]

axes_size = [610,460]

auto_resize = "on"

viewport = [0,0]

figure_name = "Графическое окно %d"

...

Свойство **children** – это массив дочерних объектов. В данном случае он включает один объект – оси координат текущего графика, обращаться к нему нужно как **h.children(1)**.

Для обращения к свойствам осей используют запись через точку. Например, так можно изменить размер шрифта (свойство **font_size**):

```
--> h.children(1).font_size = 4
```

Размер шрифта здесь задается в относительных единицах, размер 0 соответствует 8 типографским пунктам, а размер 11 – 54 пунктам. Чтобы увидеть все свойства осей, нужно ввести команду

```
--> h.children(1)
```

С помощью одной команды **plot** (а также команды **plot2d**) можно построить несколько линий на одном поле. Для этого при вызове функции перечисляются пары аргументов-массивов:

```
--> plot( x1, y1, x2, y2 )
```

Первая линия будет показывать зависимость **y1** от **x1**, а вторая – зависимость **y2** от **x2**. Массивы в каждой паре должны быть одинаковой длины.

При желании можно указать цвета для каждой линии, Например,

```
--> plot( x1, y1, "b", x2, y2, "g" )
```

Первая линия (зависимость **y1** от **x1**) будет синей, вторая (зависимость **y2** от **x2**) – зелёной. Можно использовать следующие цвета:

b – синий (*blue*)

g – зеленый (*green*)

r – красный (*red*)

c – голубой (*cyan*)

m – пурпурный (*magenta*)

y – желтый (*yellow*)

k – черный (*black*).

По умолчанию первая линия – синяя, вторая – зеленая и т.д. в порядке перечисления цветов в списке. Дополнительно можно указать тип линии:

- сплошная
- : точечная
- . штрихпунктирная
- штриховая

Например,

```
--> plot( x1, y1, "b:", x2, y2, "g--" )
```

Первая линия – точечная синего цвета, вторая – штриховая зеленого цвета. По умолчанию все линии сплошные.

Для оформления графика также используются команды

title – заголовок графика

xlabel – название оси абсцисс

ylabel – название оси ординат

У всех этих команд обязателен один аргумент – текст в апострофах или в кавычках.

Например:

```
--> title( "Импульсная характеристика" )
```

```
--> xlabel( "t, c" )
```

```
--> ylabel( "w(t)" )
```

Команда **legend** служит для вывода легенды графика. Легенда нужна, если на графике есть несколько линий и надо показать, что обозначает каждая из них. Параметрами

команды **legend**

являются символьные строки, их должно быть столько, сколько построено линий. Например, если на одном поле строятся три графика, легенда может выглядеть так:

```
--> legend( "y1", "y2", "y3" )
```

В надписях можно использовать некоторые команды системы TEX.

TEX – система подготовки текстов с математическими формулами, разработанная математиком и программистом Дональдом Кнутом. Часто используются так называемые макропакеты (надстройки), расширяющие возможности ядра TEX, например LaTeX или AMSTEX. Например, греческие буквы записываются в виде «\alpha»,

«\beta» и т.д. Верхний индекс (степень) обозначается знаком «^», Например, a^2 запишется как «a^2». Для обозначения индекса используют нижнее подчеркивание, например, a_{22} кодируется как «a_{22}». Если в надписи используются обозначения TEX, в начале и в конце нужно ставить знак \$:

```
--> ylabel( "$\phi$, градусы$" )
```

В SciLab есть возможность строить несколько графиков в одном окне. Иначе говоря, рисунок можно представить в виде матрицы: разбить на «клетки», в каждой из которых строится отдельный график. Для этого надо сделать активным нужный рисунок и применить команду **subplot**, например:

```
--> subplot( 2, 1, 1 )
```

Первое число в команде **subplot** показывает количество «строк» в такой матрице, второе – количество столбцов, третье –какой по счету график сделать активным (считая по строкам, справа налево и сверху вниз). Все дальнейшие графические команды (**plot**, **title**, **xlabel**, **ylabel**, **legend** и др.) относятся к этому «подграфику».

Для построения нашего графика можно создать новое окно и вызвать функцию

plot:

```
--> scf()
```

```
--> plot( out.time, out.values )
```

Результат вывода приведён на рис. 2.

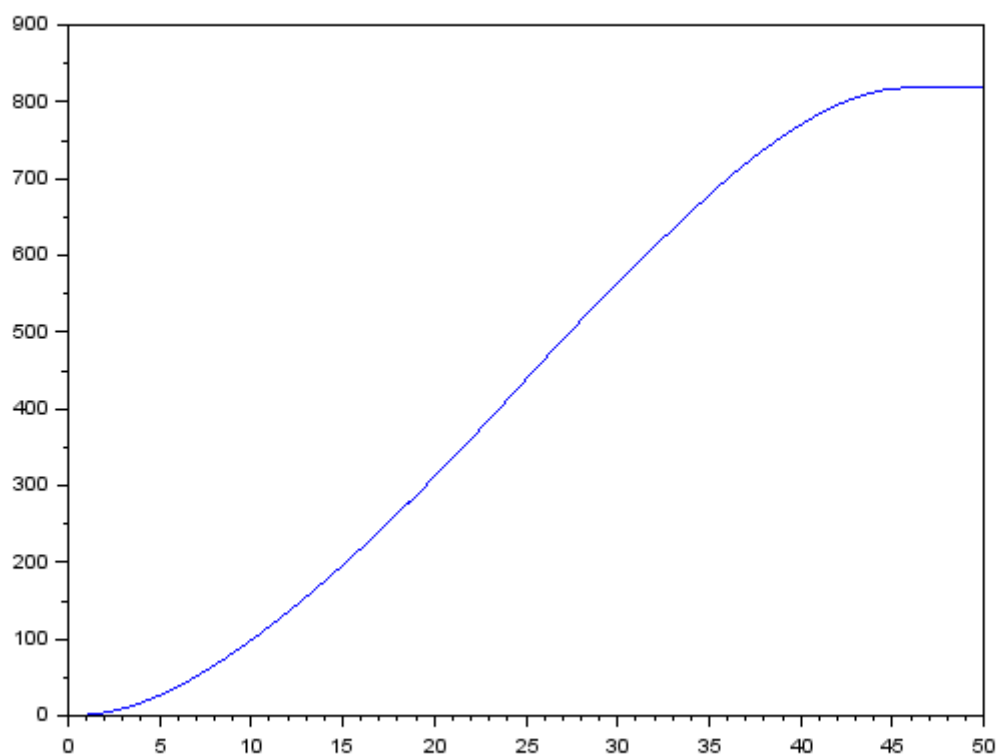


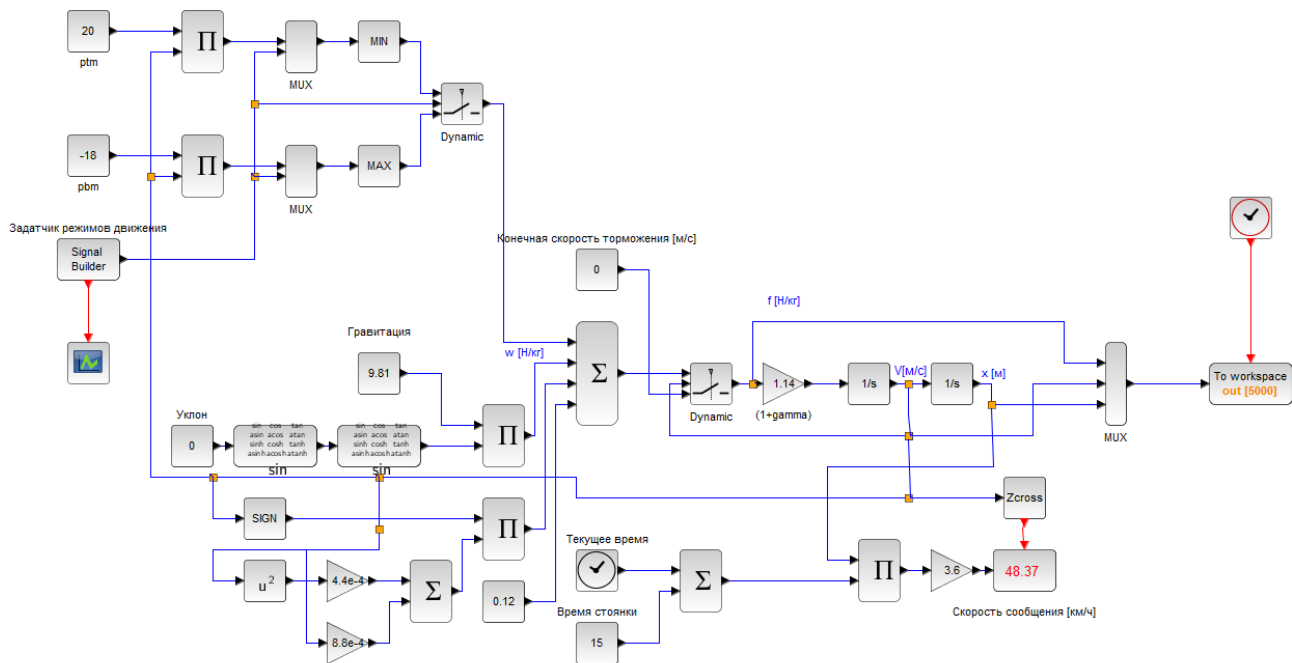
Рис. 2.

Для оформления графика можно использовать все приёмы, описанные выше.

Если нужно передать в рабочую область данные о нескольких сигналах, можно добавить в модель новые блоки *To workspace*.

Возможен и другой вариант – объединить несколько сигналов в «шину» с помощью мультиплексора и подать на вход блока *To workspace* такой многоканальный сигнал.

Добавим в модель мультиплексор с тремя входами и подадим на его входы все требуемые выходные сигналы системы, как показано на рис. 3.



Блок-схема модели движения электромобиля в условном цикле движения с учётом инерции вращающихся масс и ограничений по мощности и расчётом скорости сообщения

Рис. 3.

Выполним моделирование и проверим переменную **out**:

```
--> out
out =
```

```
values: [4999x3 constant]
time: [4999x1 constant]
```

Выясняется, что теперь поле **values** – это массив размером 4999.3, то есть он имеет 4999 строк и три столбца. Первый столбец – это значения сигнала выхода системы (с первого входа мультимплексора), второй – значения сигнала со второго выхода, а третий с третьего. К столбцам массива **out.values** можно обращаться как **out.values(:,1)**, **out.values(:,2)** и **out.values(:,3)**, соответственно.

Построим тройной график. Для этого используем команду `subplot`:

```
--> scf()
--> subplot( 3, 1, 1 )
--> plot( out.time, out.values(:,1) )
--> subplot( 3, 1, 2 )
--> plot( out.time, out.values(:,2) )
--> subplot( 3, 1, 3 )
--> plot( out.time, out.values(:,3) )
```

Handle of type "Figure" with properties:

```
=====
children: "Axes"
figure_position = [200,200]
figure_size = [626,580]
axes_size = [610,460]
auto_resize = "on"
viewport = [0,0]
figure_name = "Графическое окно %d"
figure_id = 0
info_message = ""
color_map = matrix 32x3
pixel_drawing_mode = "copy"
```

```
anti_aliasing = "off"
immediate_drawing = "on"
background = -2
visible = "on"
rotation_style = "unary"
event_handler = ""
event_handler_enable = "off"
user_data = []
resizefcn = ""
closerequestfcn = ""
resize = "on"
toolbar = "figure"
toolbar_visible = "on"
menubar = "figure"
menubar_visible = "on"
infobar_visible = "on"
dockable = "on"
layout = "none"
layout_options = "OptNoLayout"
default_axes = "on"
icon = ""
tag = ""
```

```
--> subplot(3,1,1)
```

```
--> plot(out.time,out.values(:,1))
```

```
--> subplot(3,1,2)
```

```
--> plot(out.time,out.values(:,2))
```

```
--> subplot(3,1,3)
```

```
--> plot(out.time,out.values(:,3))
```

Построенные графики показаны на Рис. 4.

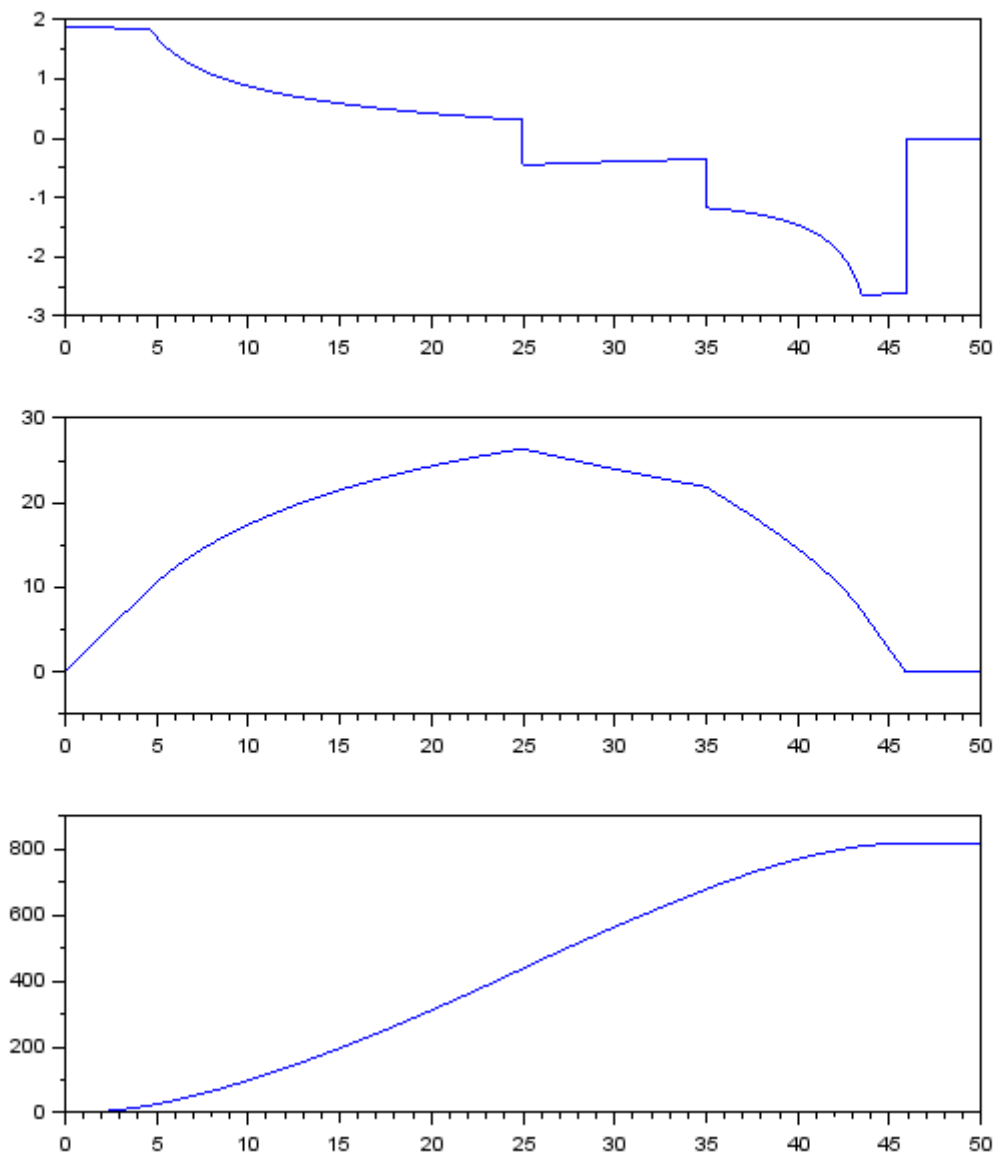


Рис. 4.

ОТЧЕТ О ВЫПОЛНЕНИИ ЗАДАНИЯ

Отчет о выполнении задания должен содержать:

1. Титульный лист.
2. Формулировку задания.
3. Блок-схему модели.
4. Анализ и описание полученных результатов.
5. Выводы.