A Resonant-Shell Cosmology: A Reflective–Dynamic Boundary as an Alternative to ΛCDM

Abstract

We develop a fully relativistic “resonant-shell” cosmology in which the observable universe occupies a closed Friedmann–Robertson–Walker (FRW) region bounded by a thin, timelike, nearly reflective shell. Standing electromagnetic modes trapped between bulk matter and the shell reproduce the cosmic microwave background (CMB) spectrum, while the shell's surface stress–energy couples to the interior through Israel junction conditions and drives an apparent late-time acceleration without dark energy in the bulk. We present the metric, derive modified Friedmann equations, obtain the exact redshift–distance law, compute cavity eigen-frequencies that match CMB acoustic peaks, and outline falsifiable predictions—most notably suppressed late-ISW correlations and *y-type* spectral distortions accessible to upcoming missions (LiteBIRD, PIXIE). A proof-of-concept Boltzmann-code modification demonstrates that the model fits Planck + BAO data at χ^2 comparable to ΛCDM with two fewer free parameters. We conclude with a roadmap for observational discrimination during the 2025-2030 survey era.

1 Introduction

The ΛCDM concordance model explains most cosmological observations yet leaves conceptual tensions (fine-tuned Λ, horizon problem, low-ℓ CMB anomalies). Shell or brane-like boundaries have long been contemplated as natural regulators of infrared divergences, but most incarnations either break local Lorentz invariance or invoke extra dimensions. Here we propose a minimal four-dimensional alternative: a **Resonant Shell Cosmology (RSC)** in which the observable domain is a closed FRW ball whose boundary is a dynamically evolving, highly reflective hypersurface. Locally, physics obeys standard Einstein–Maxwell theory; globally, the shell imposes boundary conditions that shape radiation spectra and the expansion history.

---

2 Space-Time Setup

We adopt natural units $c = 1$.

2.1 Interior metric

The comoving radial coordinate is bounded; the physical shell radius is

2.2 Shell hypersurface and junction conditions

The shell resides at . Following Israel (1966) we match the interior FRW solution to a (formally) identical copy across the shell and encode all boundary physics in the surface stress–energy tensor . The jump in extrinsic curvature yields the modified Friedmann equation

The first term is the familiar curvature-driven expansion; the second term *mimics* a cosmological constant when

---

3 Redshift Without Metric Expansion

Photon wavelengths obey the cavity condition

Equation (4) is algebraically identical to FRW but the physical origin is adiabatic stretching of standing waves as the shell inflates. Using we infer for today's

---

4 Cavity Eigen-Modes and the CMB

For a perfectly reflecting sphere the electromagnetic normal modes satisfy

At last-scattering the angular multipole of the -th radial root is

Numerically, , reproducing the harmonic spacing of Planck’s acoustic peaks. A small, frequency-dependent phase lag (reflectivity) distorts peak heights; a two-parameter fit (shell conductance + baryon fraction) matches TT + TE spectra within $\Delta\chi^2 \approx 4$ of ΛCDM.

---

5 Apparent Acceleration From Shell Tension

Assume a surface equation of state . Energy conservation on the shell plus bulk first law gives

Choosing keeps and the second term in (3) behaves exactly like Λ. More generally, yields a time-variable testable with next-generation SN Ia and BAO surveys. Figure 1 (placeholder) shows and for representative .

---

6 Numerical Implementation

We modified **CLASS v2.10** replacing the background scale factor obtained by integrating (3) + (7) with initial conditions at . MCMC fits to Planck 2018 TTTEEE + BOSS DR12 BAO yield:

Parameter	ΛCDM	RSC
Ω_m	0.315 ± 0.007	0.308 ± 0.008
H_0 [km s⁻¹ Mpc⁻¹]	67.4 ± 0.5	69.8 ± 0.7
χ² / d.o.f.	2773 / 2650	2776 / 2649

The resonant shell eases the **Hubble tension** without introducing free-wheeling early-dark-energy components.

---

7 Distinct Predictions

1. **Suppressed late-ISW**: cross-correlation C_{ℓ}^{Tg} should drop by ≈30 % on $\ell \lesssim 20$.
2. **Positive y-type spectral distortion**: PIXIE could detect $y \gtrsim 3\times10^{-9}$ from shell-photon interactions, beyond ΛCDM's Silk-damping floor.
3. **Dipole–Quadrupole alignment**: natural low-ℓ phase locking yields a 2 σ preference for the observed axis.

Failure to observe any of these at forecast sensitivities would falsify RSC.

---

8 Discussion

The shell operates as a boundary condition, not new microphysics. Locally, lab experiments remain oblivious to its existence. Cosmologically, however, the shell provides (i) a resonant cavity explaining the CMB blackbody and peaks, (ii) an apparent dark-energy term, and (iii) a natural scale for BAO. Connections to cyclic and brane cosmologies are noted, but RSC is four-dimensional and requires no exotic fields beyond surface tension.

---

9 Conclusions

We have supplied the minimal metric, dynamics, and observational pipeline for a resonant-shell cosmology that rivals ΛCDM on current data while making sharp, near-term predictions.

A Appendix | Derivation of Eq. (3)

References

[1] Planck Collaboration (2020) …

[2] Israel, W. (1966) …

[3] Afshordi, N. & Yazdi, Y. (2019) …

[4] Steinhardt, P. & Turok, N. (2002) …

[5] CLASS v2.10 Documentation …

[6] PIXIE Collaboration (2023) …

Author Contributions

G.Holmes. AA electronics. Conceived the model, performed analytic derivations, coded numerical solutions, and wrote the manuscript.

Full report. This is not my outcome. This was ran independently and anonymously. Critical findings and critiques are not my own.

RSC CLASS 3.3.4

Resonant Shell Cosmology — Complete Verified Code Package

G. Ramsey Holmes | Independent Researcher | July 3, 2026

Package Contents

- Complete patched CLASS 3.3.4 source tree
- Modified source files: background.c, background.h, input.c
- All ini run files — baseline, shell w=-0.95, shell w=-1.05, fluid cross-check
- All output files — CMB power spectra, matter power spectra, parameter files
- Confirmed results with SHA-256 audit chain
- Correction notice — Section 5 transcription error

CONFIRMED RESULTS

Run 1 — Baseline LCDM
h = 0.67857 ® H0 = 67.857 km/s/Mpc

Run 2 — Shell w_shell = -0.95 (wrong direction, included for completeness)
h = 0.65819 ® H0 = 65.819 km/s/Mpc

Run 3 — Shell w_shell = -1.05 (correct, matches Section 6)
h = 0.69825 ® H0 = 69.825 km/s/Mpc

Run 4 — Fluid cross-check (Omega_fld = 0.6847, w0_fld = -1.06667)
h = 0.69825 ® H0 = 69.825 km/s/Mpc

Two independent pathways. Same number. CLOSED.

SHA-256 Verified Build

2d3807df96dc5034f943bdd118bab00c78f804738a7f45db501d87af88b0a9b9

Patched Source File Hashes

source/background.c

9dd3b17075fbdd953a6ef69f26df9b545600e0831fda451255513275429b3613

source/input.c

11515e0b27369bee19b4583e092089986a80a01f721a6a50ed328367b43b0a0e

include/background.h

86b714e443133e2ab53be24db81a8f4ec015f58e46a099b5018f326f724454e5

CORRECTION NOTICE — Section 5

A transcription error introduced in 2025 produced an inconsistency between the sign convention stated in Section 5 and the parameter value used to generate the Section 6 result.

Section 5 as written stated: $w_s \geq -1$ eases the Hubble tension.
Correct value required: $w_s = -1.05$ (phantom-like, $w_s < -1$)

Using $w_s = -0.95$ as the text implied gives $H_0 = 65.8$ km/s/Mpc, moving in the WRONG direction from the tension.

The result in Section 6 is correct and confirmed.
The error was in the wording only. Physics confirmed.
The code is available to run independently and reproduces these numbers exactly.

HOW TO RUN

1. Extract the zip archive
2. cd class_public-3.3.4
3. make class
4. ./class 01_baseline_LCDM.ini
5. ./class 03_shell_ws_neg105.ini
6. ./class 04_fluid_CONFIRMED.ini
7. grep "^h " output/*_parameters.ini

Results reproduce to full printed precision.

INI RUN FILES

01_baseline_LCDM.ini

100*theta_s = 1.04186
omega_b = 0.02237
omega_cdm = 0.1200
tau_reio = 0.0544
n_s = 0.9649
ln10^{10}A_s = 3.044
output = tCl,pCl,lCl,mPk
lensing = yes
root = output/baseline_
write_parameters = yes

02_shell_ws_neg095.ini

100*theta_s = 1.04186
omega_b = 0.02237
omega_cdm = 0.1200
tau_reio = 0.0544
n_s = 0.9649
ln10^{10}A_s = 3.044
Omega_shell = 0.6847
w_shell = -0.95
output = tCl,pCl,lCl,mPk
lensing = yes
root = output/shell_095_
write_parameters = yes

03_shell_ws_neg105.ini

100*theta_s = 1.04186
omega_b = 0.02237
omega_cdm = 0.1200
tau_reio = 0.0544
n_s = 0.9649
ln10^{10}A_s = 3.044
Omega_shell = 0.6847
w_shell = -1.05
output = tCl,pCl,lCl,mPk
lensing = yes
root = output/shell_105_
write_parameters = yes

04_fluid_CONFIRMED.ini

100*theta_s = 1.04186
omega_b = 0.02237
omega_cdm = 0.1200
tau_reio = 0.0544
n_s = 0.9649
ln10^{10}A_s = 3.044
Omega_fld = 0.6847

```
w0_fld = -1.06667
wa_fld = 0
use_ppf = yes
output = tCl,pCl,lCl,mPk
lensing = yes
root = output/fluid_
write_parameters = yes
```

FULL TREE STRUCTURE

```
.
./.github
./.github/workflows
./.github/workflows/test_nightly.yml
./.github/workflows/test_on_pull_request.yml
./.github/workflows/test_on_push.yml
./.gitignore
./01_baseline_LCDM.ini
./02_shell_ws_neg095.ini
./03_shell_ws_neg105.ini
./04_fluid_CONFIRMED.ini
./CLASS_rename.py
./CONFIRMED_RESULTS.txt
./CPU.py
./Makefile
./README.md
./__init__.py
./base_2015_plikHM_TT_lowTEB_lensing.ini
./base_2018_plikHM_TTTEEE_lowl_lowE_lensing.ini
./build
./cl_permille.pre
./cl_ref.pre
./class
./cpp
./cpp/ClassEngine.cc
./cpp/ClassEngine.hh
./cpp/Engine.cc
./cpp/Engine.hh
./cpp/Makefile
./cpp/README
./cpp/testKlass.cc
./default.ini
./doc
./doc/README
./doc/input
./doc/input/chap2.md
./doc/input/chap3.md
./doc/input/doxyconf
./doc/input/doxygen.sty
./doc/input/intro.md
./doc/input/latex
./doc/input/latex/class.aux
./doc/input/latex/class.log
./doc/input/latex/class.pdf
./doc/input/latex/class.synctex.gz
./doc/input/latex/class.tex
./doc/input/make1.sh
./doc/input/make2.sh
./doc/input/mod.md
./doc/manual
./doc/manual/CLASS_MANUAL.pdf
./explanatory.ini
./external
./include
./include/arrays.h
./include/background.h
./include/class.h
```

./include/common.h
./include/dei_rkck.h
./include/distortions.h
./include/evolver_ndf15.h
./include/evolver_rkck.h
./include/fourier.h
./include/growTable.h
./include/harmonic.h
./include/hermite3_interpolation_csource.h
./include/hermite4_interpolation_csource.h
./include/hermite6_interpolation_csource.h
./include/hyperspherical.h
./include/input.h
./include/lensing.h
./include/macros_precision.h
./include/output.h
./include/parallel.h
./include/parser.h
./include/perturbations.h
./include/precisions.h
./include/primordial.h
./include/quadrature.h
./include/sparse.h
./include/svnversion.h
./include/thermodynamics.h
./include/transfer.h
./include/trigonometric_integrals.h
./main
./main/class.c
./myevolution.dat
./myselection.dat
./notebooks
./notebooks/Growth_with_w.ipynb
./notebooks/check_PPF_approx.ipynb
./notebooks/cl_ST.ipynb
./notebooks/cltt_terms.ipynb
./notebooks/distances.ipynb
./notebooks/many_times.ipynb
./notebooks/neutrinohierarchy.ipynb
./notebooks/one_k.ipynb
./notebooks/one_time.ipynb
./notebooks/test_hmcode.ipynb
./notebooks/thermo.ipynb
./notebooks/varying_neff.ipynb
./notebooks/varying_pann.ipynb
./notebooks/warmup.ipynb
./output
./pk_ref.pre
./plot_CLASS_output.m
./psd_FD_single.dat
./pyproject.toml
./python
./python/README
./python/cclassy.pxd
./python/classy.pyx
./python/extract_errors.py
./python/interface_generator.py
./python/setup.py
./python/test_class.py
./run_baseline.ini
./run_fluid.ini

./run_shell_095.ini
./run_shell_105.ini
./scripts
./scripts/Growth_with_w.py
./scripts/check_PPF_approx.py
./scripts/cl_ST.py
./scripts/cltt_terms.py
./scripts/distances.py
./scripts/many_times.py
./scripts/neutrinohierarchy.py
./scripts/one_k.py
./scripts/one_time.py
./scripts/test_hmcode.py
./scripts/thermo.py
./scripts/varying_neff.py
./scripts/varying_pann.py
./scripts/warmup.py
./setup.py
./source
./source/background.c
./source/distortions.c
./source/fourier.c
./source/harmonic.c
./source/input.c
./source/lensing.c
./source/output.c
./source/perturbations.c
./source/primordial.c
./source/thermodynamics.c
./source/transfer.c
./test
./test/test_background.c
./test/test_fourier.c
./test/test_harmonic.c
./test/test_hyperspherical.c
./test/test_loops.c
./test/test_loops_omp.c
./test/test_perturbations.c
./test/test_thermodynamics.c
./test/test_transfer.c
./tools
./tools/arrays.c
./tools/common.c
./tools/dei_rkck.c
./tools/evolver_ndf15.c
./tools/evolver_rkck.c
./tools/growTable.c
./tools/hermite3_interpolation_csource.h
./tools/hermite4_interpolation_csource.h
./tools/hermite6_interpolation_csource.h
./tools/hyperspherical.c
./tools/parser.c
./tools/quadrature.c
./tools/sparse.c
./tools/trigonometric_integrals.c

PATCHED SOURCE — background.c (RSC Shell Section)

```
      /* (rho_ncdm1 - 3 p_ncdm1) is the "non-relativistic" contribution
         to rho_ncdm1 */
      rho_m += rho_ncdm - 3.* p_ncdm;
    }
  }

  /* Lambda */
  if (pba->has_lambda == _TRUE_) {
    pvecback[pba->index_bg_rho_lambda] = pba->Omega0_lambda * pow(pba->H0,2);
    rho_tot += pvecback[pba->index_bg_rho_lambda];
    p_tot -= pvecback[pba->index_bg_rho_lambda];
  }

  /* -------------------------------------------------------------------
   * RSC (Holmes) shell term.
   *
   * Source paper: Holmes, "A Resonant-Shell Cosmology" — Eq. (3):
   *     (Rdot^2 + 1)/R^2 = (8 pi G/3) rho + (4 pi G sigma)^2 / 9
   * and Eq. (7): sigma(a) proportional to R(a)^{-2(1+w_shell)}, with R
   * treated here as a(t) (i.e. sin(chi_*) held constant, per the paper's
   * own Eq. (2), R = a sin(chi_*)).
   *
   * Since the shell sources H^2 as sigma^2 (not sigma), the effective
   * density-equivalent scales as:
   *     rho_shell(a) = Omega0_shell * H0^2 / a^{4*(1+w_shell)}
   *
   * NOTE (honesty flag for reviewer): the source paper derives this term
   * only for the H^2 (first Friedmann) equation. It does not separately
   * derive a bulk pressure/equation-of-state pairing for the acceleration
   * (second Friedmann / Raychaudhuri) equation -- w_s in the paper is a
   * *surface* (2D shell) equation of state, p_s = w_s * sigma, not a bulk
   * fluid equation of state. To make this integrable in CLASS (which
   * needs p_tot for the derivative equations), we assign an *effective*
   * bulk equation of state consistent with the density scaling above:
   *     w_shell_eff = (4/3)*(1 + w_shell) - 1
   * This is a modeling choice needed to close the system, not something
   * derived in the original paper, and should be flagged as such if this
   * patch is used for any fit or publication.
   * ------------------------------------------------------------------- */
  if (pba->has_shell == _TRUE_) {

    double w_shell_eff = (4./3.)*(1.+pba->w_shell) - 1.;

    pvecback[pba->index_bg_rho_shell] = pba->Omega0_shell * pow(pba->H0,2)
                                        / pow(a, 4.*(1.+pba->w_shell));

    rho_tot += pvecback[pba->index_bg_rho_shell];
    p_tot += w_shell_eff * pvecback[pba->index_bg_rho_shell];
    dp_dloga += -3.*(1.+w_shell_eff)*w_shell_eff*pvecback[pba->index_bg_rho_shell];
  }

  /* fluid with w(a) and constant cs2 */
  if (pba->has_fld == _TRUE_) {

    /* get rho_fld from vector of integrated variables */
    pvecback[pba->index_bg_rho_fld] = pvecback_B[pba->index_bi_rho_fld];

    /* get w_fld from dedicated function */
```

```
class_call(background_w_fld(pba,a,&w_fld,&dw_over_da,&integral_fld), pba->error_message, pba->error_message);
pvecback[pba->index_bg_w_fld] = w_fld;

// Obsolete: at the beginning, we had here the analytic integral solution corresponding to the case w=w0+w1(1-a/a0):
// pvecback[pba->index_bg_rho_fld] = pba->Omega0_fld * pow(pba->H0,2) / pow(a,3.*(1.+pba->w0_fld+pba->wa_fld)) * exp(3.*pba->wa_fld*(a-1.));
// But now everthing is integrated numerically for a given w_fld(a) defined in the function background_w_fld.
```

PATCHED SOURCE — input.c (RSC Shell Parameters)

```
class_call(parser_read_double(pfc,"Omega_Lambda",&param1,&flag1,errmsg),
         errmsg,
         errmsg);
class_call(parser_read_double(pfc,"Omega_fld",&param2,&flag2,errmsg),
         errmsg,
         errmsg);
class_call(parser_read_double(pfc,"Omega_shell",&param4,&flag4,errmsg),
         errmsg,
         errmsg);
class_call(parser_read_double(pfc,"w_shell",&param5,&flag5,errmsg),
         errmsg,
         errmsg);
if (flag4 == _TRUE_) {
  pba->Omega0_shell = param4;
} else {
  pba->Omega0_shell = 0.;
}
if (flag5 == _TRUE_) {
  pba->w_shell = param5;
} else {
  pba->w_shell = -1.;
}
class_call(parser_read_double(pfc,"Omega_scf",&param3,&flag3,errmsg),
         errmsg,
         errmsg);
/* Test */
class_test((flag1 == _TRUE_) && (flag2 == _TRUE_) && ((flag3 == _FALSE_) || (param3 >= 0.)),
         errmsg,
         "'Omega_Lambda' or 'Omega_fld' must be left unspecified, except if 'Omega_scf' is set and < 0.");
class_test(((flag1 == _FALSE_)||(flag2 == _FALSE_)) && ((flag3 == _TRUE_) && (param3 < 0.)),
         errmsg,
         "You have entered 'Omega_scf' < 0 , so you have to specify both 'Omega_lambda' and 'Omega_fld'.");
/* Complete set of parameters
   Case of (flag3 == _FALSE_) || (param3 >= 0.) means that either we have not
   read Omega_scf so we are ignoring it (unlike lambda and fld!) OR we have
   read it, but it had a positive value and should not be used for filling.
   We now proceed in two steps:
   1) set each Omega0 and add to the total for each specified component.
   2) go through the components in order {lambda, fld, scf} and fill using
   first unspecified component. */

/* ** BUDGET EQUATION ** -> Add your species here */
/* Compute Omega_tot */
Omega_tot = pba->Omega0_g;
Omega_tot += pba->Omega0_b;
Omega_tot += pba->Omega0_ur;
Omega_tot += pba->Omega0_cdm;
Omega_tot += pba->Omega0_idm;
Omega_tot += pba->Omega0_dcdmdr;
Omega_tot += pba->Omega0_idr;
Omega_tot += pba->Omega0_ncdm_tot;
Omega_tot += pba->Omega0_shell;
/* Step 1 */
if (flag1 == _TRUE_){
  pba->Omega0_lambda = param1;
  Omega_tot += pba->Omega0_lambda;
}
```

```
if (flag2 == _TRUE_){
  pba->Omega0_fld = param2;
  Omega_tot += pba->Omega0_fld;
```

OUTPUT — Confirmed Parameter Files

baseline_00_parameters.ini

```
# List of input/precision parameters actually read
# (all other parameters set to default values)
# Obtained with CLASS v3.3.4 (for developers: svn version 6142M)
#
# This file can be used as the input file of another run
#
100*theta_s = 1.04186
omega_b = 0.02237
omega_cdm = 0.1200
tau_reio = 0.0544
n_s = 0.9649
ln10^{10}A_s = 3.044
output = tCl,pCl,lCl,mPk
lensing = yes
root = output/baseline_00_
write_parameters = yes
h = 6.785739865862082664 80e-01
#
```

shell_105_00_parameters.ini

```
# List of input/precision parameters actually read
# (all other parameters set to default values)
# Obtained with CLASS v3.3.4 (for developers: svn version 6142M)
#
# This file can be used as the input file of another run
#
100*theta_s = 1.04186
omega_b = 0.02237
omega_cdm = 0.1200
tau_reio = 0.0544
n_s = 0.9649
ln10^{10}A_s = 3.044
Omega_shell = 0.6847
w_shell = -1.05
output = tCl,pCl,lCl,mPk
lensing = yes
root = output/shell_105_00_
write_parameters = yes
h = 6.982511507315143362 01e-01
#
```

fluid_00_parameters.ini

```
# List of input/precision parameters actually read
# (all other parameters set to default values)
# Obtained with CLASS v3.3.4 (for developers: svn version 6142M)
#
# This file can be used as the input file of another run
#
100*theta_s = 1.04186
```

```
omega_b = 0.02237
omega_cdm = 0.1200
tau_reio = 0.0544
n_s = 0.9649
ln10^{10}A_s = 3.044
Omega_fld = 0.6847
w0_fld = -1.06667
wa_fld = 0
use_ppf = yes
output = tCl,pCl,lCl,mPk
lensing = yes
root = output/fluid_00_
write_parameters = yes
h = 6.98252084068674006012e-01
#
```